AGARD-CP-504

AGARD-CP-504

AGARD

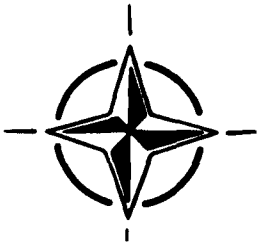
ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT
7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD CONFERENCE PROCEEDINGS 504

Air Vehicle Mission Control and Management

(La Gestion et le Contrôle des Missions des Véhicules Aériens)

Papers presented at the Guidance and Control Panel 53rd Symposium held at the Marine Kazerne, Amsterdam, The Netherlands from 22nd October to 25th October 1991.



NORTH ATLANTIC TREATY ORGANIZATION

Published March 1992

Distribution and Availability on Back Cover

AGARD-CP-504

AGARD

ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT

7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD CONFERENCE PROCEEDINGS 504

Air Vehicle Mission Control and Management

(La Gestion et le Contrôle des Missions des Véhicules Aériens)

Papers presented at the Guidance and Control Panel 53rd Symposium held at the Marine Kazerne, Amsterdam, The Netherlands from 22nd October to 25th October 1991.

North Atlantic Treaty Organization
Organisation du Traité de l'Atlantique Nord

The Mission of AGARD

According to its Charter, the mission of AGARD is to bring together the leading personalities of the NATO nations in the fields of science and technology relating to aerospace for the following purposes:

- Recommending effective ways for the member nations to use their research and development capabilities for the common benefit of the NATO community;
- Providing scientific and technical advice and assistance to the Military Committee in the field of aerospace research and development (with particular regard to its military application);
- Continuously stimulating advances in the aerospace sciences relevant to strengthening the common defence posture;
- Improving the co-operation among member nations in aerospace research and development;
- Exchange of scientific and technical information;
- Providing assistance to member nations for the purpose of increasing their scientific and technical potential;
- Rendering scientific and technical assistance, as requested, to other NATO bodies and to member nations in connection with research and development problems in the aerospace field.

The highest authority within AGARD is the National Delegates Board consisting of officially appointed senior representatives from each member nation. The mission of AGARD is carried out through the Panels which are composed of experts appointed by the National Delegates, the Consultant and Exchange Programme and the Aerospace Applications Studies Programme. The results of AGARD work are reported to the member nations and the NATO Authorities through the AGARD series of publications of which this is one.

Participation in AGARD activities is by invitation only and is normally limited to citizens of the NATO nations.

Published March 1992

ISBN 92-835-0662-6

Printed by Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ

Theme

Mission control and management is becoming an increasingly important subject in vehicle guidance and control. Developments in this field provide crucial contributions to the improvement of mission effectiveness. These are related to single air vehicle missions or to multiple air vehicle missions (including air traffic), and to the whole missions or the mission elements.

The subject covers both the ground segment and the air vehicle segment of the complete system, with highly integrated concepts embracing ground-based and onboard system elements, being particularly relevant. The ground segment typically comprises a number of elements such as pre-mission tasking and planning, ground-based information sources, situation assessment and mission plan upgrading. The air vehicle segment comprises the onboard mission control and management, including situation assessment capabilities, automatic mission planning and plan execution functions. The onboard system may be used as an aid to the pilot in mission planning and tactical decision making or, in the case of an unmanned vehicle, may be fully automated.

The technical papers addressed the current developments and trends in air vehicle mission management, assessed the state-of-the-art and identified technological alternatives for future systems. System application issues are of primary importance. The following topics fell within the scope of the symposium: air vehicle missions, situation assessment technology, mission planning and execution, system implementation for mission control/management.

Thème

La gestion et le contrôle des missions revêt de plus en plus d'importance dans le guidage et le pilotage des véhicules aériens. Les développements dans ce domaine représentent des contributions d'une importance capitale pour l'optimisation de la conduite des missions. Ils concernent des missions soit à vecteur unique, soit à vecteur multiple (y compris le trafic aérien) et ils s'appliquent soit à la totalité de la mission, soit à certains éléments de celle-ci.

Le sujet couvre le segment sol et le segment véhicule aérien du système complet, où la conception hautement intégrée des éléments du système, tant embarqués qu'au sol, est particulièrement significative. Le segment sol comprend typiquement un certain nombre d'éléments tels que l'assignation des tâches et la planification avant vol, les sources d'informations au sol, l'évaluation de la situation et la mise à jour du plan de mission. Le segment véhicule aérien comprend la gestion et le contrôle de la mission par des moyens aéroportés, y compris l'évaluation de la situation, la planification automatique de la mission et l'exécution du plan de mission. Le système aéroporté peut soit servir d'aide au pilote pour la planification de la mission et la prise de décisions tactiques, soit, dans le cas d'un véhicule non-piloté, fonctionner en mode automatique.

Les communications techniques ont examiné les réalisations et les tendances actuelles dans le domaine de la gestion des missions des véhicules aériens, présenté l'état de l'art et proposé des solutions technologiques pour la conception des systèmes futurs. Une attention toute particulière a été accordée aux questions concernant les applications. Le symposium a porté sur les sujets suivants: les missions des véhicules aeriens, les techniques d'évaluation de la situation, la planification et l'exécution de la mission, et la mise en oeuvre des systèmes de gestion/contrôle des missions.

Guidance and Control Panel

Chairman: Dr E.B. Stear
Corporate Vice President
Technology Assessment
The Boeing Company
PO Box 3707
Mail Stop 13—43
Seattle, WA 98124-2207
United States

Deputy Chairman: Mr S. Leek
8 Sunnyfield
Hatfield
Hertfordshire AL9 5DX
United Kingdom

TECHNICAL PROGRAMME COMMITTEE

Chairman:	Prof. Dr Ing. R.C. Onken	GE
Members:	IPA L. Guibert	FR
	Dr H. Winter	GE
	Dr B. Mazzetti	IT
	Ir G. Alders	NE
	Dr P. Sanz-Aranguez	SP
	Mr J. Simon Calero	SP
	Mr A.D. King	UK
	Mr J.K. Ramage	US

HOST NATION COORDINATOR

Captain (Rtd) Ir L. Sombroek
c/o National Aerospace Laboratory (NLR)
Netherlands Delegation to AGARD
P.O. Box 126
2600 AC Delft
The Netherlands

PANEL EXECUTIVE

Commandant M. Mouhamad, FAF

Mail from Europe:
AGARD—OTAN
Attn: GCP Executive
7, rue Ancelle
F-92200 Neuilly sur Seine
France

Mail from US and Canada:
AGARD—NATO
Attn: GCP Executive
Unit 21551
APO AE 09777

Tel: 33(1) 47 38 57 80
Telex: 610176 (France)
Telefax: 33(1) 47 38 57 99

ACKNOWLEDGEMENTS/REMERCIEMENTS

The Panel wishes to express its thanks to the Dutch National Delegates to AGARD for the invitation to hold this meeting in Amsterdam and for the facilities and personnel which made the meeting possible.

Le Panel tient à remercier les Délégués Nationaux des Pays Bas près l'AGARD de leur invitation à tenir cette réunion à Amsterdam et de la mise à disposition de personnel et des installations nécessaires.

Contents

* Published in classified volume CP-504 (Supplement).
† Not available at time of printing.

* Published in classified volume CP-504 (Supplement).

KEYNOTE ADDRESS

by

Major-General W. Breeschoten
Ministerie van Defensie (KLu)
Luchtmachstaf
P O Box 20703
2500 Es Den Haag
The Netherlands

I. INTRODUCTION

Ladies and Gentlemen,

In addition to my present function as Director Operations of the RNLAF I am a fighter pilot.

This week you will be discussing my job; the management and control of missions.
Therefore I was pleased, both with the subject of this symposium and with your kind invitation to address you at the beginning of this week.

In this address I will start with my views as a pilot on the subject of mission control support systems, (you already noted the word support) and work towards my views as Director. These views are the same, only the scope is different.

II. MISSION CONTROL AND THE PILOT

As a pilot, or group of pilots I get assigned a mission, say to attack a certain ground target with the objective to destroy it, or make it unusable for a certain period of time. My job then is to prepare and execute this mission.

My preparation consists largely of building up an image of what precisely is required and under what circumstances.
For this I collect information on the situation enroute, the situation at the target, the assistance that can be expected from supporting forces and when, the weather situation and so on.

Then the mission is planned, which concerns a multitude of items:
route selection, flight profile planning, weapons selection and attack profile planning, planning for mutual protection , planning for ECM employment, planning for threat system counters, not to mention planning for relative timing and required command and information exchanges with my companions with supporting units and with other missions.

Now I know what I can expect during my mission and how to manage possible situations. Also I know how these situations can be identified, using my on-board sensor data. For situations that can be anticipated I know what I am supposed to do and how to do it.

During execution I use the preparation results in conjunction with my system displays and my mkl eyeball to maintain an image of the situation, *assess its meaning for my mission*, decide on actions and of course carry them out.

This process, from preparation to execution, is called mission control. In the complex technical and operational environment of today and tomorrow I can do with some help when doing it.

First of all in the field of information during preparation. The more I know, the less surprises, and the larger the chance of a successful mission. Therefore information must be recent, reliable, and preferably accurate and complete. Most of all it must be easy to interpret.
For instance, I want to know what my target looks like now, not last week; where precisely it is or is expected to be, and what tricks can be expected in terms of decoys camouflage, defenses etc.

Then, when planning my mission, I certainly like support in clerical tasks.
Let a system calculate my fuel use for me, plot my route, identify the risk when engaging threats for the tactics selected, document my planning for me and prepare my system data loader for me.
Let it allow me to assess the consequences of an info update at the end of my planning process and make it possible to adapt the primary plan if necessary.

During my mission I would like my systems to show me the situation in an easily understood manner. Show me relevant information only. I could indicate premisssion what I consider relevant and what not. It can also show me my own capabilities in relation to threats.

You will notice that I stress the information supply side, and the improvement of the quality of information, to make it directly accessible for the pilot, without the need to interpret raw data.
I do not want the system to think for me, at least not in the sense that it prescribes my tactical actions.
It can to some extent think with me. For example to carry out some tasks that I do not wish to be bothered with; it may monitor my other systems, decide which ones do not perform correctly, re-configure, and inform me of any degradation in performance, not redundancy.
Also, if I want so, a system can perform specific tasks for me, for instance do my self-protection, activate and shut down jammers and throw decoys, provided its operation is transparent, which means that I know what it will be doing.

More in general, the functions of systems that support me need to be clear. I have to understand what their products mean.
For route planning I like a system that shows me the known threat, and indicates what possibly could be there that is unknown at present.
I certainly do not want a system that calculates an optimum route for me under the erroneous assumption that the world is known, with the result that I have to meander towards my target and then run into a few surprises all the same.

It is imperative that mission control support for manned systems is designed in such a manner thatit supports the pilot, or more in general the operator. If he can not maintain a clear image of the situation, he cannot perform his job. Also, if the support systems do perform part of his primary job, he does lack the freedom of control he may need. And if you let the system do it all, you do not need a pilot.

III. MISSION SUPPORT SYSTEM DEVELOPMENT

The key word above is support to the user. To be able to do this, the user must be known. System designers may think they know the user. Human factor engineers may think they know the user, the user may think he knows himself. They all are wrong.
Know yourself. Most people need more then a lifetime to do only that. The user does not know what he needs until he is confronted with what he thought he needed, and again and again. For all the others it is the same.

Recognition of this fact has guided the development of mission support systems in the Netherlands. The mission preparation system CAMPAL is an example. The development of this system has been gradual. It has been rebuilt three times, and been evaluated each time at one or more operational squadrons. Presently its software is the core of MSS/C, where C stands for CAMPAL, which is being developed for EPAF F-16 users by GD. You will be briefed later this week on this system.

A similar approach is taken with the development of an Air Base Command, Control and Information System. At present a part of the system is operational on one of our air bases, as a stage in its development. I should mention in this respect, that its development is presently under review, mainly because of the recent geopolitical changes. The old concept, that most operations would be from the own MOB's, which is one of the elements in the ABCCIS design, seems to be no longer valid for the RNLAF.
I will address this point later.

IV. FUTURE ENVIRONMENTS FOR MISSION SUPPORT SYSTEMS

As you all noticed, the world has changed. The WP threat that dominated military thinking in the West has disappeared. The Soviet threat has changed in nature. In the wake of the fall of the WP society new situations where military actions could be necessary emerge. This is affecting military thinking, concepts of operation, required means and mission contents.
It also will affect the requirements for systems that support the missions.

I will try to sketch the broad context.

The open question is where, against whom, with whom, and under what rules the next conflict will have to be fought. It is not only a question of how we assemble a fighting force, but even more so how we realize the required effective command, control and information supply.

Whereas in the recent past the dominating drive was the threat from the east with a limited number of possible scenarios, present developments show a larger variation in these scenarios, which also are less well defined. How to cope with that?

The buzzword used here is versatility. The terms you encounter are immediate reaction forces, rapid reaction forces and so on.

The recent conflict in Iraq is just one example of such a possible scenario.

What did we see?
Advanced technology assets such as stealth aircraft were used for initial strikes to gain advantage and create air superiority. Medium range cruise missiles were used to attack high value fixed targets. Lethal and functional defense suppression was used successfully at a large scale, and the bulk of the airborne forces employed medium altitude tactics. Low level tactics were employed mainly where the weapon characteristics dictated its use. We saw large scale use of precision guided weapons, to avoid unwanted damage, and a hustle for damage assessment, the latter even becoming some sort of concern, since the level of detail needed could not be achieved with long range sensors.

For this specific case lessons have been learned. Viewed in the light of the new NATO situation the lessons seem to have the following shape:

In the Gulf scenario medium altitude operations were predominant. Does it mean low level tactics are out? Certainly not. Still one should be prepared to do it. Next time the opponents defense could be more potent, and some time may be needed to clear the skies. The main lesson is that if you do not need low level tactics, do not use it. Choose the one appropriate for the situation.
It means that air crews and systems, including mission control support for manned airborne weapon systems should be able to cope with both environments.

The conflict also showed the benefit of being able to operate around the clock. In the near future technology will enable operations at night for an even larger proportion of weapon systems. Again a variation in operation that require mission control support systems to be versatile.

The one dominating aspect in the Gulf conflict has been the ability to hit the proper target and probably even more important, not to hit the wrong target. This faculty is the main factor in securing political and public support during the conflict. Target identification power and weapon delivery accuracy are the main ingredients of the recipe.

Target identification is not only a matter of separating the friend from the foe, but even more so the identification of the non-combatant. Systems thus should support positive identification of friend, non-combatants and foes.
It is equally important in the air-to-air as in the air-to-ground situation. If you fail to identify an erring airliner on the wrong track and you shoot it down, it does not help your own mental state, nor the political status of the conflict.
Inversely, if your system does not allow positive identification, you may not be able to use your fine BVR capability, and be forced into a more risky short range exchange. If you shoot up a convoy of refugees, instead of a tank column, it does not help a lot to state that those guys had no business there.

This is not a new problem, it only has become even more important. Much effort has been spent over the years to try to improve the situation. Efforts vary from networked solutions, where an overall picture is generated centrally, using and

interpreting information from available sources, and distributing the result by data-link, to non-cooperative target recognition methods on-board weapon systems.
There is no simple solution to this problem. All available information should be combined in assessing who is who. The final step still remains to be done on-board. Here, pre-mission and beamed-up information have to be combined with the information from on-board sensors to make the final assessment. Truly a fine task for mission control support systems.

The present command and control concept works with specialised units that are tasked from a high level. Specialisations include defense suppression, escorts for bombing missions, pre-strike recce and BDA recce. The operation as a whole is planned in a rather large detail, resulting in a massive flow of tasking orders. It seems that certainly in transitional stages, i.e. during the build up at the beginning of a conflict, the availability of these specialised units is at a premium, and the coordination process is cumbersome. Therefore the use of so called composite forces, where the necessary support functions are available under one commander, at one location, should be seriously considered.

With composite forces, the tasking can be simpler. Also an increased effectiveness of such forces can be expected, since these can easily train together. But the local effort in mission planning and management will increase. Composite force actions could be regarded as super-missions, a fact that will undoubtedly affect the desirable supporting systems properties.
Composite Forces or not, for many Air Forces, certainly for the Dutch, the old situation, where the main action was expected to take place from the own MOB's is no longer valid. More likely next actions will be from bases elsewhere, with services and command and control provided by others, together with nationals that speak other languages, but with whom you need to cooperate and coordinate. Thus, the preferred properties of planned and already realised infrastructure on the own MOB's related to mission preparation and execution do change. Such facilities will need to be deployable, or be available at the host base, when wartime use is envisaged. The other part will have primarily a training value. Moreover, the question what parts of the command and control system needs to be standardized comes into a new light. At first sight, it seems appropriate that mission preparation and mission management support systems will remain weapon-system specific for quite a long time to come. The main argument for this is that the weapon-systems themselves will be quite different.
These supporting systems thus have to be deployable. For the casual observer, the command, control and information systems that supply the mission systems with tasking and information could well be standardized. Meteo info, intelligence info, tasking orders, availability data etc. do lend themselves to standardisation in information contents. The process to arrive at technical and information standards is under way, but by no means completed.
The properties of mission control support systems will affect the standardisation agreements that are being developed, and vice versa. The definition of missions, think about the concept of composite forces, will also affect the functions of these systems.

Being technical people, your main angle of incidence this week will be systems and enabling technologies.
You will address methods to harness the flow of data, and transfer it into information. You will address methods to structure the control process, so that it will be amenable to analysis, and you will exchange information on your experiences while developing systems.
The properties of these systems and even of elements of these products are closely related to the structure of the world in which these are to be used. The opposite is also true, this user world is affected to a large extend by the possibilities of technology. Therefore, in my opinion there is an important contribution to be made by AGARD in the field of the definitions of standards, both on the technical and on the information level.

V. CLOSING REMARKS

At the end of my address I like to re-iterate the three elements that constitute the main body of my speech, in reversed order.

In your work, try to contribute to the definition of adequate standardisation rules, that allow the necessary variety in operations and in systems, while stimulating effectiveness.

When considering mission control support systems, allow for versatility. Do not assume fixed scenarios. Make no "optimum solutions" for incomplete worlds.

And last but not least, when designing for manned systems, remember the pilot. Think about his mission and about what he really needs. Allow for use of human faculties. Keep him informed and keep him in the loop. If not, make a drone.

As you may have guessed, I think the subject of this symposium is highly relevant. It is good to see that so many scientists take an interest in it. Undoubtedly, the symposium will spur many discussions on the subject and thus stimulate your creativeness into new ideas. I wish you all an interesting, pleasant and fruitful week.

INTERDICTION MISSION PLANNING AND CO-ORDINATION DURING THE GULF CONFLICT 17 JAN - 27 FEB 91

V.A. MEE
P.J. GOODMAN
COMBAT OPERATIONS CENTRE
ROYAL AIR FORCE BRUGGEN
BFPO 25
LONDON

SUMMARY

The Gulf Air War started for Royal Air Force Tornado GR1 aircraft with the delivery of JP 233 weapons from low level at night. After sustaining unacceptable losses at low level, tactics changed to delivery of freefall 1000 lb bombs from medium level, at first by night and then by day. Accuracies, however, improved dramatically with the arrival in theatre of precision guided munitions.

The paper will describe a typical mission covering all aspects from initial tasking by ATO, through mission planning and coordination with intelligence input to deconfliction and mission execution. Missions were flown as co-ordinated packages, Tornado GR1's being supported by F15C fighter escorts with EW support from F4G "Wild Weasels" and EF111 "Raven" aircraft. Mission planning by the Royal Air Force was manpower intensive and could have been streamlined with the availabilty of more automated systems.

Several lessons were learned from the conflict, which are listed and discussed in the text.

1 INTRODUCTION

This paper discusses Interdiction Mission Planning and Co-ordination of RAF Tornado GR1s in the Gulf during Operation Granby/Desert Storm. It will cover the phases of the air war, the planning and execution of a typical interdiction mission and some of the lessons learned from the operations in the Gulf. The participation of the Tornado consisted of 3 Squadrons of GR1s, one each at Dhahran and Tabuk in Saudi Arabia and one at Muharaq in Bahrain. There were also a Flight of GR1As in the reconnaissance role at Dhahran and a Flight of GR1s in the ALARM role at Tabuk. Later in the conflict, a Flight of Tornados equipped with the TIALD pod also operated from Tabuk.

2 PHASES OF THE AIR WAR

The air war can be divided into 4 phases:

a. Low level attacks on enemy airfields using JP 233 and lofted 1000 lb bombs in the airburst mode to suppress defences.

b. Night medium level 1000 lb bombing.

c. Day medium level 1000 lb bombing.

d. Precision bombing with laser designation by day and by night.

Phase 1 - Low Level Attacks. From the start of the war on 17 Jan 91 for 4 nights Tornado GR1's were engaged in low level attacks on enemy airfields, using JP 233 for runway denial and lofting (or tossing) 1000 lb freefall airburst bombs in an effort to suppress airfield defences. JP 233 was generally effective, going for angled cross-cuts on runways and taxiways. However, the Iraqi main operating bases were so large, with so much redundancy of operating surface, that in many cases the best tactic was to plan to cut off access from the HAS sites to the runways. Lofting 1000 lb on to defences was not as effective as had been hoped, as accuracies achieved were not good enough to significantly reduce AAA. In fact they often had a negative effect, being the first weapons to detonate in the target area, they generally alerted the gunners on the airfield, who started a AAA barrage before aircraft dropping JP 233 arrived. Iraqi AAA defences were much heavier than anticipated and it seemed that every man who had a gun had been ordered to fire it into the air as soon as an incoming raid was detected. There appeared to be few problems with ammunition suuply and it was quite literally like flying into a wall of lead. The bomber force was sustaining unacceptable losses, which forced the next phase.

Phase 2 - Night Medium Level Attacks. On 21 Jan, after losing 5 Tornados, tactics were changed to night, medium-level, freefall 1000 lb bombing, against the enemy's main airfields, large petro-chemical plants and ammunition dumps. By this time the enemy's Early Warning and defence co-ordination systems had been severely degraded and, apart from some ineffective attacks by Mirage F1s against F-111s, his fighters has shown no real indications of joining the battle. However, there was major concern about SAM's and AAA at the chosen operating altitude of around 20,000 ft, which was in the middle of the threat envelopes for SAM 2, 3, 6 and the Improved Hawk, and on the margins of SAM 8 and Roland. Although only the heaviest calibre AAA could reach above 18,000 ft, several SAM concentrations were positioned to funnel aircraft into heavy calibre AAA "killing boxes".

To counter the SAM threats , the Tornado had a comprehensive self-defence Electronic Warfare capability. The Radar Warning Reciever alerted crews to the threat, the Skyshadow jamming pod possessed counters to many of the SAM systems and the BOZ pod enabled crews to dispense chaff and IR decoy flares. In addition, the USAF had concentrated its worldwide SEAD assets into a relatively small theatre of operations and support from EF-111 "Ravens", F4G "Wild Weasels" and F15-C fighter aircraft was always available. These support aircraft were integrated with the bombers to form consolidated packages.

The effects of the night bombing campaign were difficult to assess since timely Bomb Damage Assessments (BDA) were not easily available, unless the target was an oil refinery which tended to flare magnificently when hit.

Phase 3 - Day Medium Level Attacks. By 3 Feb, it was assessed that the fighter threat was minimal and the SAM threat was decreasing, therefore, a switch was made to daylight medium level operations. Using shallow dive attacks still from medium level, the crews were able to employ visual bombing techniques and get ranging sensors like the laser and the ground mapping radar (GMR) locked on to targets. Accuracy improved and specific targets on airfields, like stockpiles of munitions, hangars aircraft in the open, were attacked.

Phase 4 - Laser Guided Bombing. It was only with the introduction of laser designators into the RAF's inventory in the Gulf that the accuracy of weapon delivery really improved. The Squadrons at Tabuk dropped precision guided munitions (PGM) designated by Tornados equipped with two pre-production Thermal Imaging Airborne Laser Designator (TIALD) pods whilst at Dhahran and Muharaq the Buccaneers designated with the Pave Spike laser pods and the GRIs dropped Pave Way laser guided bombs (LGB). Now crews were able accurately to target bridges (including pontoon bridges hastily erected to replace the ones that had already been knocked down), individual HASs and runway intersections. The TV imaging also provided an instant assessment of the results of the attacks and crews did not have to return to targets which had already been hit.

3 A TYPICAL MISSION PACKAGE

In describing a typical mission package, one of 87 flown by GR1s out of Dhahran, the following points will be covered:

tasking and the intelligence input, mission planning, co-ordination and deconfliction, and the final intelligence input before the mission is flown.

The mission, No. 0301G, was an 8-ship of Tornado GR1s tasked against Ar Ramadi Petroleum Store, some 40 miles to the West of Baghdad. Time on Target (TOT) was 2300z on 31 Jan, so the mission was a night medium level sortie, each aircraft armed with 8x1000 lb freefall bombs. Initial target notification was by secure telephone and fax from Joint Air Headquarters at Riyadh, about 36 hours before TOT. Formal notification was received with the arrival of the Air Tasking Order (ATO) at about 0100z on the day of the attack. The ATO was the typical US document of approximately 500 computer-generated pages, similar those with which RAF aircrew who have attended exercises like Red Flag, have become familiar. It detailed the support package for the bombers and this consisted of 4xF-15C as fighter escorts, 4xF4G "Wild Weasels" and 2xEF-111 "Ravens" as EW support. In addition were details of which AWACS would provide airborne control and, equally important, the air-to-air refuelling plan for the raid. Great distances were involved; for this sortie, a track distance of some 1600 miles was flown, which required the Tornados to tank both outbound and inbound.

Once the target had been plotted, mission planners put together an approximate route based on the tanker plan from the ATO and minimum risk routes from the Airspace Co-ordination Order (ACO). The latest information about SAM, AAA, radar and Early Warning sites was provided by intelligence specialists. Planning was carried out using the Feranti CPGS Mk III system which can process flight planning information but has only a limited ability to process intelligence. There was no shortage of intelligence and an accurate picture of threat locations was built up and maintained. A very close working relationship evolved between the RAF and the USAF in intelligence matters. The RAF was given access to Sentinel Bite and Constant Source, and crews were occasionally able to use MSS II for confirming the route and attack profiles which offered the best chance of success and the least danger from threats.

Detailed planning of the route and the attack was always carried out by the aircrew who would be flying the mission. However, assistance was provided by the mission planning staff who supplied fixpoints and suitable offsets for radar bombing from information available in the fixpoint catalogues and basic target graphics produced by the Joint Air Intelligence Reconnaissance Centre (JARIC). If sufficient information was not available, the mission planners would mensurate new fixes and offsets by using the Point Positioning System (PPS) and very accurate 1:10,000 acetate overlays of the targets provided by the Directorate of Military Survey at Feltham, West London. This assistance was very labour-intensive and could have been facilitated by continual access to threat assessment systems such as MSS II which were available to USAF squadrons.

The ATO deconflicted targets in the same area by the use of time blocks. Detailed deconfliction and co-ordination within the package was worked out between the Package Commander and his element leaders by using the American supplied secure telephone system, KY 68. Without this common system it is doubtful that package co-ordination and integrity could have been achieved. Over to the West of Saudi Arabia, the Tabuk squadrons got most of their package support from the Red Sea Carrier Group (the USS Kennedy and Saratoga). Liaison with the US Navy was difficult because there was no common secure voice system and all routing and timings had to be passed to the US Navy Liaison Officer at Riyadh, who relayed the information to the carriers over the Navy network, a system fraught with delays and scope for error.

Deconfliction within the package was generally accomplished by timing and height separation as shown in Fig 1. As the package flew North from the border at a groundspeed of 450kt, the Weasels and Ravens were about 2 minutes ahead of the Tornados. The Tornados were at about 22-24000 ft with the Ravens below them at about 20-21000 ft. The Weasels were at 25-29000 ft and the fighter escort ahead and above at about 30,000 ft.

Throughout the ingress, the Weasels acted as forward observers and augmented the Tornados' RHWRs by calling any threats they picked up. The Ravens provided continuous jamming of any threat radars. As the formation crossed 32°30' North, two of the F-15s peeled off to the north-east to block any fighters taking off from airfields around Baghdad. Approaching the IP, the Weasels and Ravens accelerated away to set up CAPs 15 miles West of the target to start "working" the target area threats, with the remaining F-15s providing top-cover. Shortly afterwards, the Weasels claimed a HARM kill on a SA3 site and at the IP they continued to give the Tornados information on threats still radiating in the target area. However, the very presence of the Weasels and the Ravens acted as a deterrent to the enemy defence systems. With all the support elements in place as shown in Fig 2, the Tornados flowed through the target at 15-20 secs spacing on two attack tracks, releasing at about 4 miles to go. The last aircraft called off-target, allowing the package to reform, climb and egress to the South.

All 8 Tornados released their weapons on the target, the crews reported seeing massive explosions and a great deal of unaimed AAA coming up to about 15,000 ft, which seemed to increase in intensity as each set of bombs impacted. The target was left burning and fires could still be seen from the air 80 miles away.

4 LESSONS LEARNED

The package system is the most effective use of limited assets and concentrates the available forces in time and space. For this reason, if UK forces become in any future conflict, it would be best to prosecute the war in alliance with US forces. To that end, the RAF should retain its familiarity with US tactics, procedures and with the ATO type of tasking. RAF participation in exercises such as Red Flag, Maple Flag and the Composite Air Operations (COMAO) package training system used in the Central Region should be increased.

A common, secure communications system is essential for package planning and deconfliction.

Within the RAF, intelligence and mission planning are manpower dependant. More automated systems, like the MSS II, which link intelligence and planning, need to be developed. It is understood that Hunting is developing such a system, Pathfinder 2000, for use with the Harrier GR5/7 and the tactical transport fleet. This or a similar automatic system, should be made available to the Tornado force so that routing, fixing, targetting and intelligence information can be displayed and manipulated quickly and accurately. The complexity of modern weapons systems means that future operations will always be mounted from fixed bases with technical back-up facilities. Therefore, it should be planned to take advantage of these facilities by using sophisticated computer systems to reduce the workload and manpower in such areas as mission planning and intelligence dissemination.

Weapons need to be versatile. A weapon that can only be delivered from low level becomes so much scrap metal when the delivery tactic changes to medium level. The JP 233 and the BL 755 fell into this category, leaving only the freefall 1000 lb bomb as a viable weapon for the Tornado.

Finally, the most important lesson, that tactics must remain flexible. Upon discovering that enemy airfields were heavily defended by AAA, the Tornados were forced to abandon low level attacks and to bomb from medium level, a role for which the RAF was totally unprepared because for 25 years it had been practising to fight a low level offensive air war. Medium level bombing proved to be very effective in this theatre when smart weapons were eventually used, but it could have been more effective earlier had aircrew previously trained in that environment. Had it been necessary and had the Iraqis started using their airfields again, there is no doubt that the Tornados would have returned to low level to deliver JP 233. Delivery tactics, though, would have been different from the large co-ordinated formation attacks originally used. Single aircraft or pairs flying through simultaneously at low level and high speed at night, seperated by long and random gaps, would have retained the advantage of surprise and hopefully delivered the weapons before the AAA barrage had been alerted.

5 CONCLUSION

The 4 phases of the air war for the Royal Air Force in the Gulf evolved through changing tactics in response to a changing threat. Missions were flown as packages, the bombers recieving fighter and EW support from the USAF. Accurate and up to date intelligence information was always available from a variety of sources and mission co-ordination was simplified by the availability of good secure telephone systems. The main lessons learned from the conflict were as follows:

- Force packaging is the most effective use of limited assets and should be regularly practised by NATO aircrews.

- A common secure communications system is essential for package planning and deconfliction.

- The Tornado force should aquire an automatic mission planning system capable of displaying the latest intelligence picture and planned track in order to select the minimum risk route.

- Tactics should remain flexible and planners must not fall into the trap of thinking that an offensive air war can only be fought from low level thus leaving a gap in our arsenal of medium level munitions. A stand-off runway and area denial weapon should be developed without delay.

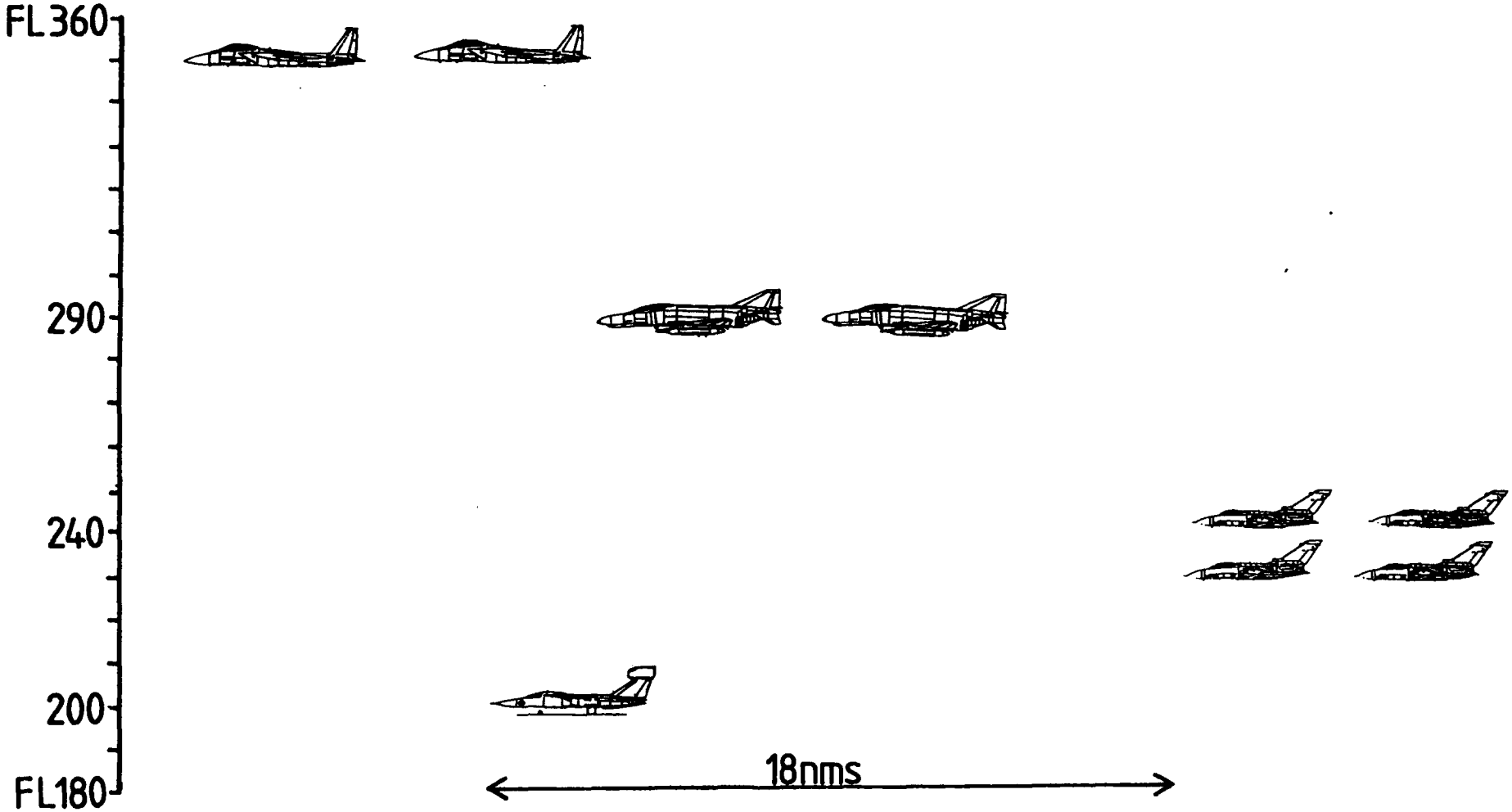


Fig 1. Package Profile

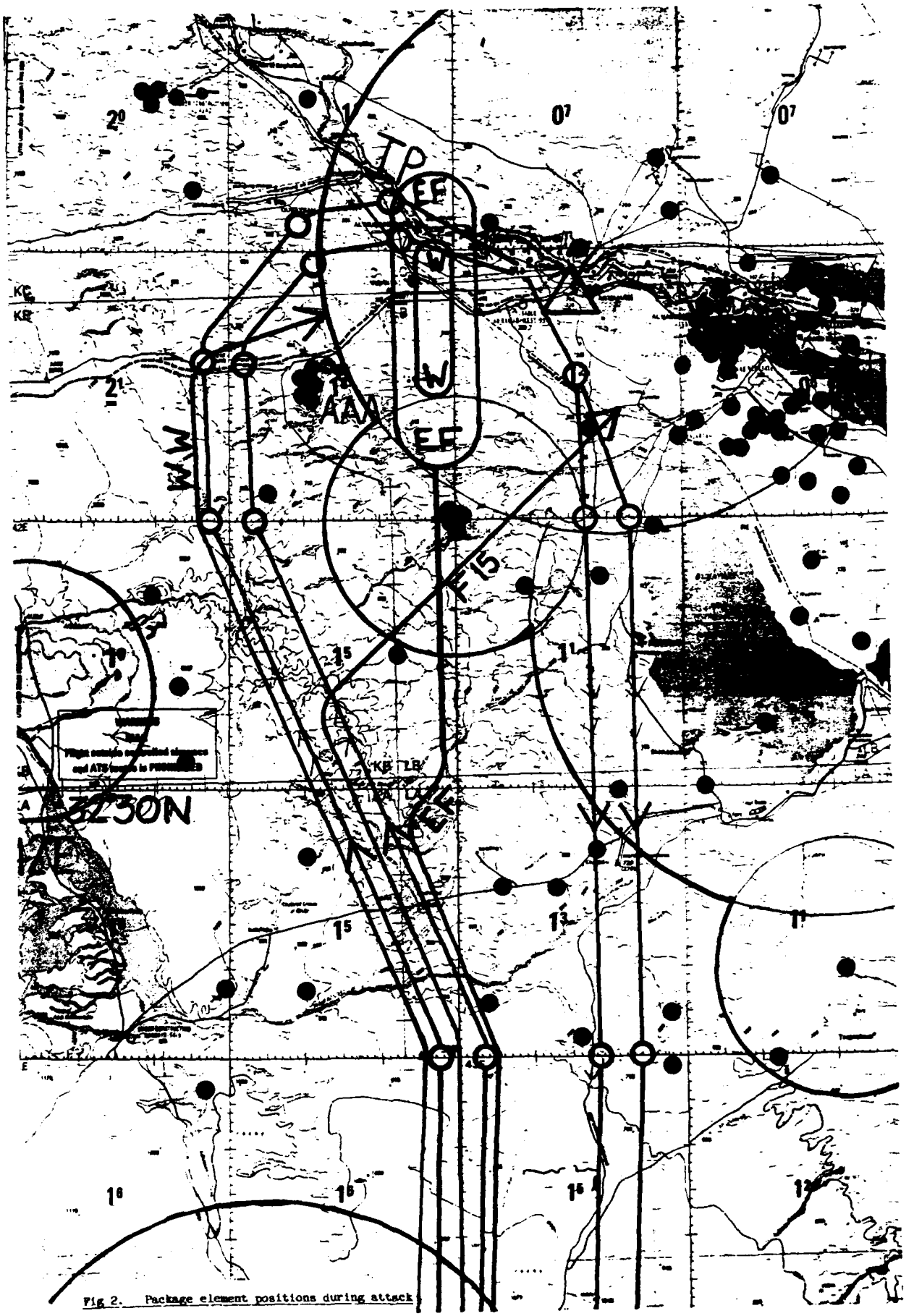


Fig 2. Package element positions during attack

AirPower Planning and Battle Management Training at Airbase Level

P. Schulein, R.F.W.M. Willems
Netherlands Defence Research Organisation,
Physics and Electronics Laboratory TNO,
P.O. Box 96864, 2509 JG,
The Hague, The Netherlands

1. SUMMARY

Airborne or Ground-Based Pilot Mission Planning is a single important link in the chain of activities that is required to facilitate operating airpower from an airbase. Other elements of that chain include Target Recce and Allocation and Sortie Generation.

It turns out that information management problems associated with intelligence generation, also apply to effectively controlling Sortie Generation, both to make most efficient use of scarce resources and to generate an organisation that is able to benefit from Pilot Mission Planning improvements.

An obvious solution in improving an information management problem is designing a Command, Control Communications and Information System (C3IS). To improve design and development of these C3I systems, it is proposed to combine the design of those systems and the use of Battle Management Training Systems.

Developing operational and training systems together has possibilities to increase user involvement and acceptance and incorporate studies towards not only just fielding a C3I system, but looking at the most effective combination of System and Organisation.

As a case the design of the Airbase Operations Wargame for the Royal Netherlands Airforce is presented and some conclusions are drawn regarding the possible use of this system in designing C3I systems such as the Airbase Command and Control Information System ABCCIS.

2. INTRODUCTION

During the last few years, and especially towards and during the Gulf-war a lot of effort has been directed towards Pilot Mission Planning Tools. The requirements of flexible response on targets of opportunity, the incorporation of real time intelligence, the mission packaging and a decrease in available resources has lead to a requirement to shorten Pilot Mission Planning times and to increase Mission Planning Quality. A number of Pilot Planning Tools have been designed and fielded.

But Pilot Mission Planning is only a link in the chain of projecting Airpower:

1. Target Recce
2. Target Prioritizing
3. Target Allocation
4. Mission Planning
5. Mission Preparation
6. Mission Launch
7. Mission Execution
8. Mission Debriefing

In these steps a lot of emphasis is placed on the quality of available information.

Numerous studies have already indicated an information problem within the intelligence community. The sheer size of the available intelligence data makes it very difficult to generate adequate and timely target/mission planning information. This is being addressed by trying to design automated systems. Pilot Mission Planning is a part of the intelligence and planning process that could be supported by such systems.

However effective projection of Airpower not only precludes quality target information and efficient use of available resources, but also an exact knowledge about the status of those resources.

Projecting Military Force is not only being able to generate that force, but also being able to plan beforehand exactly when and where that generation takes place. That means for airpower planning that adequate knowledge about the Sortie Generation Capabilities is needed to insure that realistic tasking and reliable Times over Target can be realized.

Sortie Generation Capability Assessment is therefore of prime importance, not only to insure realistic allocation and tasking, but also to make efficient use of the available Sortie Generation Resources on the airbases. Increasing the quality of these assessments not only increases the quality of the airpower planning and Sortie Generation Control, but also is mandatory to make effective use of Pilot Mission Planning Aids, by increasing the effectiveness of the Sortie Generation organisation.

Obviously, increasing the quality and speed of Pilot Mission Planning without insuring a well run organisation to perform preparation and launch has no benefits for Airpower projection as a whole. All links in the Airpower Projection chain should be strenghtened equally.

In the following chapters focus will be placed on improving Airpower Planning Effectiveness and improving Sortie Generation Capability Effectiveness.

3. INCREASING AIRPOWER PLANNING EFFECTIVENESS THROUGH C3I SYSTEMS

Airpower planning is largely dependent on the availability of timely accurate information, both on targets, current situation and resource stati, and tools or methodologies to manipulate this information as needed at will. Therefore Command, Control, Communications and Information Systems (C3IS's) are designed and fielded.

A large part of the current C3I system designs are technology driven. Obviously all design methodologies that are geared to produce an accurate, logical and consistent system are mandatory in designing these complex systems.

However, as experiences in civil automation projects have shown, a system concept should not only be based on what is technologically feasible or desirable but should also be based on the best way of embedding technology in the organisations that are supposed to use these systems in order to work more effectively [Schulein (1), Borawitz (4)].

This implies that the way in which an organisation is going to use a C3IS should be

identified before the system is designed.

Any system that just delivers data without taking into account the ways the organisation and its management processes are structured is at best a Communication and Information System.

The way an organisation is going to use a C3IS is dependent on goals, tasks, means, situation, procedures, tradition, and worst of all: people. To identify the first 6 items is already a tough problem, but to measure the impact of behaviour of groups or individuals is extremely difficult. A close match between the organisational behaviour and a C3IS is required to gain acceptance and participation from the (future) users. This match can only be made by looking at the complete concept of the C3IS and the organisation together.

The ability to deliver an effective automated system without any organisational changes has proven to be almost impossible to date. So before a C3IS is designed it should be quite clear in what way the target organisation and its management is going to use or misuse the system.

This is not only mandatory to be able to design and field a system that will be accepted by the users, but also is needed to be able to validate the operational effectiveness of that C3IS. As experiences in both civil and military applications have shown, automating some activities of an organisation will not automatically mean that the effectiveness or the production of that organisation (airforces and projecting airpower) is increased.

To involve (future) users in the design and development process a way should be found to "let them play" with the concept of the system in a simple try-out. For non-technical users it is extremely difficult to express their requirements on paper to a computer expert. But most users perfectly know what they want or like when they have hands-on experience with a model.

The solution for this is hardly original: building a prototype of the system for the user to play with is a common solution.

However, a prototype in conventional means has some drawbacks. Often the prototype is ill defined, almost trivial in contents and even with modern software development techniques quite expensive to build. What really would be ideal to have is a testbed system that is cheap, reliable, especially build to be changed and redesigned, with sufficient substance to allow users to "play" with it for a longer time without becoming trivial or boring.

4. INCREASING SORTIE GENERATION EFFECTIVENESS

4.1 Activities

Sortie generation activities can be divided into four categories:

* Tasking
  Acquiring realistic tasking conditions.
* Sortie Generation
  Developing an effective organisation.
* Capability Assessment and Resource Management
  Developing an effective organisation control.
* Pilot Mission Planning
  Developing an effective mission execution.

To increase the Sortie Generation effectiveness of an airbase, measures can be taken

in each of these four categories.

4.2 Improvement Measures

The improvement of all categories is served with improvement in resource availability and sustainability. But barring this a number of other measures could be taken.

To improve Tasking the following solutions could be pursued:
- Up to date status reports
- Intelligence Control Systems
- Resource Control Systems

To make Sortie Generation execution more efficient a number of procedural or organisational measures can be taken:
- Squadron/Weaponload Standardisation
- Pre-Planning and Pre-Loading
- Resources and Training
- Interoperability

Elements of Capability Assessment and Resource Management that could be supported are:
- Assessment
- Resource Control
- Response Times
- Reporting.

And of course the Pilot Mission Planning Tools and Procedures already mentioned could also be applied.

Physical measures concerning Sortie Generation execution have been taken in the past and are still important. However to increase the effectiveness of the Sortie Generation Capabilities the control of those activities is becoming much more prominent. So what activities can be performed to increase Sortie Generation Management Effectiveness?

5. INCREASING BATTLE MANAGEMENT EFFECTIVENESS

5.1 General

Battle management at airbase level can be improved by improving the following basic activities:
* Base Capability Assessment
* Sortie Generation Assessment
* Real Time Status Reporting.

A number of measures can be taken to increase the performance of these activities:
* Standardisation of
  - Intelligence Sources
  - Reporting
  - Interoperability
* C3I Systems for:
  - Data Management
  - Decision Support
* Training at:
  - Airbase management
  - C3I system use

Basically two kinds of solutions could be pursued to increase the performance mentioned: incorporate C3I systems at airbase level and increase Training of Management Personnel at Airbase level. Designing a C3I system for an airbase encounters the same problems as presented in chapter 4.

To increase training facilities Battle Management Training Systems are needed.

5.2 Battle Management Training Systems

Battle management means the complex integration of all aspects of combat: intelligence, planning, support, execution, logistics, command and control, assessment and environment.

Together with the continuous need to assess the units capability in order to control force planning and the introduction of Command, Control and Information Systems (CCIS), the working environment of

battle managers continues to expand.

Furthermore to introduce the new tools and procedures effectively, it is necessary to review existing management structures and to examine ways to adapt these structures to evolving requirements. Due to the unavailability of exercises to look into these problems because of cost and risk, training systems must be used.

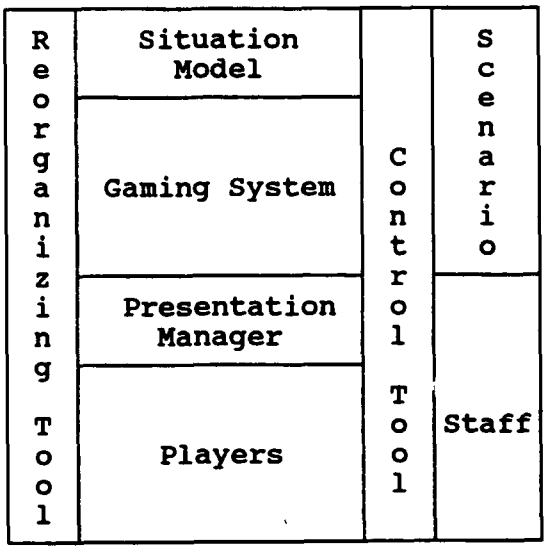
Training systems can be divided in two categories. One category offers training opportunities that can also be achieved by other means such as exercises, but that can be performed less expensive or less dangerous on simulation systems. The other category contains systems that provide training opportunities that cannot be achieved by any other training methods other than actual combat operations.

This category contains Battle Management Training Systems. In these "man in the loop" systems not only the handling of complex situations is simulated (as in wargames) but also tools are offered to review and possibly reorganize the way in which the management model of the game is implemented.

The use of automated systems and computer support turns out to be mandatory for training systems that model an organization and its activities over time. To model and present an organization and to extrapolate a current situation into the future by relating the situation to all actions that have been initiated by the trainees requires the manipulation of a huge amount of data. Real time manipulation for training purposes is beyond human capability without computer support.

So Battle Management Training Systems are in essence highly interactive and flexible computer simulations.

Due to developments in computer sciences in both hardware and software areas it became feasible to build these systems. The components of such a system typically include:



The situation model describes the actual system (for example an airbase) that is simulated.

The gaming system has the basic gaming structures that allow the situation model to be run as an interactive game.

The presentation manager generates the user interface and controls the information and command flows between the players and the gaming system.

The reorganisation tool allows the staff to change the Presentation Manager and Gaming System in such ways that other management or interface structures can be tried.

The control tool allows the staff to run the simulation.

In these BMTS's only one playing team is indicated and the

"enemy" is played by the scenario and/or staff. This is important to be able to precisely address management problems or decision moments, since in this situation the flow of the exercise can be tighter controlled than in a two sided, free play game.

Because the system inherently is built to be changed and redesigned as far as the Presentation Manager is concerned, ideas that are generated by the players using the system can be readily incorporated. Of course changing the presentation manager is largely equivalent with redesigning an albeit smaller equivalent of a C3I system! And because the Battle Management Training System is not an empty shell like most Prototyping systems, the players cannot only address presentation of information, but also availability, timeliness and operational impact. In this way a Battle Management Training System can be used as a testbed for a future C3I system.

6. USING A TESTBED C3I/ MANAGEMENT TRAINING SYSTEM

Two applications of a Battle Management Training System have emerged: on one hand it can be used to perform its primary goal of providing Battle Management Training, on the other hand it can be used as a testbed to design and develop C3I Systems.

A number of important considerations concerning the use of Training Systems this way are:

* Low Cost
Two demands of the user can be met by most efficient use of resources. The knowledge about the situation at hand that needs to be acquired to successfully build either a C3IS or a BMTS is largely the same. The user acceptance problems are largely the same. Both from a conceptual and a technical standpoint designing Battle Management Training facilities and C3I Systems are largely equivalent (i.e. application, design methodologies, technology, user interface, functionality) [Schulein (2)].

* Judgement of effectiveness
By intensive reviewing in a testbed environment more information can be gained about the benefits of the C3I system and their operational effectiveness. Be aware however that conclusions drawn from these Training Sessions need to be interpreted very carefully before using them in practice.

* User availability and involvement
As has been stated before the design and development for a C3I system or a Battle Management Training System should include not only technology but (obviously) also organisational goals and tasks. Furthermore, when a support system of any kind is added to an existing organisation a reflection on existing management structures is advisable.

It turns out to be surprisingly difficult to express the goals, tasks, means and procedures in such a way that this description can be used to design systems to improve operational and training effectiveness.

Fortunately, when Battle Management Training is introduced within the military academies and colleges those problems mentioned before are exactly the subjects that are addressed there. The officers in the training facilities, especially those facilities that address management training as opposed to procedural training, are available to rethink and restructure their management

problems more freely than officers in operational service, who are required to run a service.

These officers have time to think, therefore take them seriously, not only in their knowledge about the situation but also in their abilities to review your technical solutions.

A bonus effect emerges. Officers playing the Training system, knowing beforehand that their remarks will not only be taken seriously in respect to the Training System itself, but also will be studied in respect of their own future C3I systems, will be extremely motivated.

* Technical development
To keep the Training system manageable accept simplification in amount of data or amount of detail, but include all principles of datahandling that should be present in the C3I system. Otherwise interpretation of remarks or ideas from training sessions will be extremely difficult to use for the real system.

Make a clean division between the components mentioned before and aim for redesign of Presentation Manager, leaving the situation model intact.

Look further into ways of actually using the same hardware and software components for the training system and the C3I system.

Ensure that the final C3I system incorporates a training mode that includes more than just learning which buttons to push.

* Practice
To give you some feeling about a real case the next chapter will address the Airbase Operations Wargame.

7 WARTIME AIRBASE OPERATIONS: THE AIRBASE OPERATIONS WARGAME

7.1 General

The Royal Netherlands Airforce (RNLAF) has designed a course for the RNLAF Staff College to facilitate Battle Management training for RNLAF officers. FEL-TNO was tasked to design and develop a training aid with the primary goal to address Battle Management at airbase level

7.2 Characteristics

The Battle Management Training System that was designed, the Airbase Operations Wargame (AOW), is in service with the Royal Netherlands Airforce Staff College, it includes all aspects of operating from a main operating base: Air Operations, Sortie Generation, Logistics and Support, Infrastructure, Passive Defence and Active Air/Ground Defence [Boots (3)].

The system models the airbase organisation below the level of the management team. The users of the system play the part of management team members. Players communicate with the system using the same ways a computerized Battle Management System would interact with real life management teams [Schulein (5)].

The AOW-1 Single Player system is operational at the RNLAF Staff College. The AOW-2 Multi-Player System is under development on a PC network and is scheduled to be completed in 1992.

7.3 Lessons Learned

During the design and development of this Airbase Operations Wargame system the Physics and Electronics Laboratory was also tasked to design an Airbase Command and Control Information System

(ABCCIS) for use on the RNLAF airbases.

These projects were carried out by two different sections of the laboratory, that held close contact during the design and development of both systems.

Obviously the design for a nation wide Airbase Command and Control and Information System like ABCCIS is a highly complex task. All the same, from a user standpoint only the following important differences exist between the ABCCIS system and the Training System:

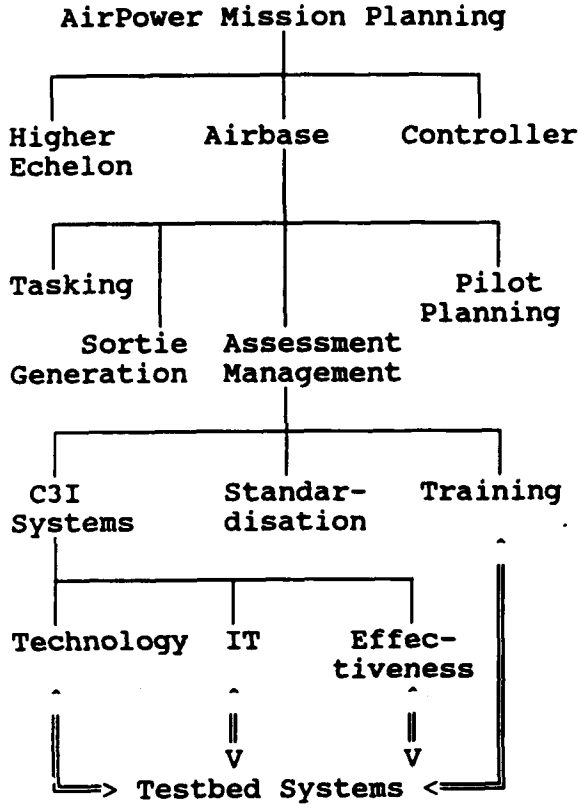
- the amount of data in the training system is limited;
- the training system will interact with a computer model instead of a real airbase
- the training system is operational at the RNLAF Staff College, the ABCCIS system exists in the design papers (status july 1991).

While using the Airbase Operations Wargame the officers playing part in the simulation management teams addressed a number of points concerning availability of information, presentation, decision aids and came up with management alternatives.

Because changes in the training system are made fairly easy and indeed the system is built to simulate different management structures, in depth discussions about the most feasible combination of management organisation and C3I Systems were stimulated. The Airbase Operations Wargame became a building box to express new ideas about Airbase Management and its C3IS's.

8 CONCLUSIONS

The following train of thought has prompted importance of using Battle Management Training Systems as testbed systems for Command Control Communication and Information Systems:



Building C3I and BMT systems together is feasible but requires precise definitions of the application area and the goals. Even with precise goals, validation of the system and ranking of the trainees will be very difficult to quantify in case of a BMTS and the impact on actual operations will be very difficult to quantify in case of a C3IS.

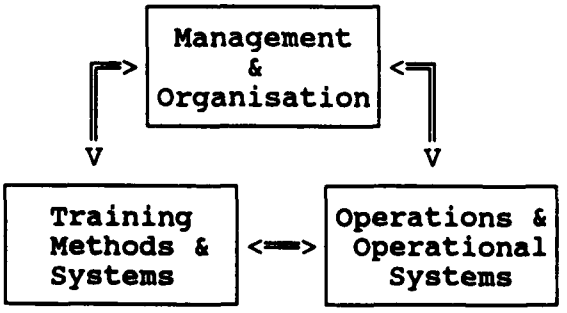
Battle Management Training Systems can be used to play testbed roles while examining possibilities of improving Battle Management Functions. Officers serving as instructor or as trainee in battle management classes can be a welcome source of ideas and feedback concerning BMTS's and

C3IS's. Further research is needed to create concepts of combining BMTS's and C3IS's into the same physical system.

When designing or developing C3IS's or BMTS's, not only procedural training or decision making should be addressed, but alternative management structures should also be considered: the organisation and its C3IS is a package deal.

The technological, methodological and design problems of BMTS's and C3IS's turn out to be equivalent. Further research should be directed in the direction of developing methods to further integrate those two systems, not only in design and use but also in physical components.

In this way the most economical combinations of C3IS's and BMTS's can be derived and even after fielding an integration between operational use and training activities can be maintained:



9. REFERENCES

1. P. Schulein, "Crisis gaming for research and training", In: Simulation-gaming: On the improvement of competence, 19th ISAGA Conference, aug 1988, Utrecht, Netherlands, Paper

2. P. Schulein, "Design, Development, and Application of a Crises Game", TNO Workshop "Human Failure in Process Industry", 1990, Paper

3. E.A.M. Boots, "User Manual Airbase Operations Wargame", FEL-TNO report FEL-91-A083.

4. W.C. Borawitz, E.A.M. Boots, P. Schulein, "Battle Management Training", The Sixth European Forum for System Simulation and Management Gaming, Paper, Nov 1990

5. P. Schulein, E.A.M. Boots, "Training Airbases at Operational Level: Airbase Operations Wargame", The Seventh International Symposium on Military Operations Research, Shrivenham, UK, September 1990, Paper

AUTOMATED TARGET TRACKING AND LOCATION TECHNIQUES APPLIED TO OPTICAL PAYLOADS ON REMOTELY PILOTED VEHICLES

Ing. Mario Audenino, Ing. Antonio Gaglio, Ing. Paolo Faggion

ALENIA SISTEMI DIFESA
DIVISIONE AVIONICA ED APPARATI SPECIALI
Strada Aeroporto
10072 Caselle Torinese (Torino), Italy

Summary
This paper illustrates procedures and techniques for automated target tracking and location. A description of the Mini-RPV System with target location analysis based on error propagation theory is given, while Video Tracker System is discussed.

Abbreviations

ADC	Air Data Computer
AG	Antennas Group
AHRS	Attitude and Heading Reference System
CCS	Command and Control System
CEP	Circular Error Probability
DAAS	Divisione Avionica ed Apparati Speciali (Avionics and Special Systems Division)
FLIR	Forward Looking Infra Red Sensor
FMS/AP	Flight Management System and AutoPilot
FOV	Field of View
GCS	Ground Control Station
GPS	Global Position System
LLLTV	Low Light Level TV camera
LOS	Line Of Sight
LRS	Launch and Recovery Subsystem
MPES	Mission Programming and Evaluation Station
MSK	Minimum Shift Keying
NRPS	Navigation Receiver Processing System
RPV	Remotely Piloted Vehicle
SRTT	Sistema di Rice-Trasmissione e Tracking (Receive-Transmit and Tracking System)
UAV	Unmanned Air Vehicle
VTTC	Video Telemetry and Telecommand

List of Symbols

θ	RPV Pitch Angle
ϕ	RPV Roll Angle
ψ	RPV Heading Angle
Az	Platform Azimuth Angle
El	Platform Elevation Angle
σ_x	x Standard Deviation
σ	Pan Angle
γ	Tilt Angle
ϕ_t	Twist Angle
h	RPV Altitude
R	Slant Range

1. Introduction

The ALENIA Mini-RPV System has the capability to perform different missions; one of the most useful missions is to provide real-time target acquisition (detection, recognition, identification and location) from the battlefield. When the target has been identified the Mini-RPV System within the GCS processes data arriving from the air vehicle and with the aid of digital terrain maps gives the target location by determining geographical coordinates and altitude. Target location is calculated using the vector from the air vehicle to the target and the RPV present position. The RPV to Target vector information relative to air vehicle body coordinate is derived from sensor LOS data acquired using a stabilized platform. This platform permits target tracking irrespective of air vehicle motion. With present on-board electronics and digital terrain maps the target location accuracy is approximately 50 meters CEP.

2. Mini-RPV System Description

The Mini-RPV System has the capability of performing a wide range of missions. A few examples of the types of missions that the MIRACH 26 can perform are briefly outlined below.

Local reconnaissance, acquisition of intelligence data concerning the activities and resources of the enemy.

Battlefield surveillance, systematic observation of enemy areas, places and activities.

Target acquisition, detection, recognition, identification and location of targets.

Artillery adjustment, acquisition of data to rapidly and accurately engage selected targets

with indirect fire weapons.
Damage assessment, acquisition of battle damage information.

The Mini-RPV System (see fig.1) comprises an air vehicle (MIRACH 26) with a choice of electro-optical payloads, a Ground Control Station (GCS), Command Control Station (CCS) plus Antennas Group (AG), an anti-jamming datalink, a Mission Programming and Evaluation Station (MPES), a Launch and Recovery Subsystem (LRS) and related Maintenance/Refurbishment Subsystem.

The air vehicle is the airborne platform containing the mission payload. The air vehicle basic elements are the air vehicle body, the avionics and the payload. The air vehicle body is partially built in composite materials and power is provided by a two stroke engine turning a pushing propeller.

The avionics installed on-board (see fig.2) include the Air Data Computer (ADC), Attitude and heading Reference System (AHRS), Navigation Receiver Processing System (NRPS), Video Telemetry and Telecommand (VTTC) and Flight Management System and Autopilot (FMS/AP).

The FMS/AP performs the following functions : mission management, navigation and related sensors management, guidance and flight control, payload command and control, data link control and flight plan management. The FMS/AP is produced by ALENIA DAAS and is currently undergoing integration flight testing.

The complete avionics configuration is achieved with the use of the Global Position System (GPS) backed by the OMEGA/VLF-Diff.system (NRPS). In this configuration the navigation accuracy is better than 50 m CEP within a radius of 30 km and gives the option of autonomous flight. The MIRACH 26 has the capability to switch from remote control to autonomous control and vice-versa. Furthermore the RPV has the capability to fly in dead-reckoning navigation mode if both the communication and navigation radio link are lost.

The on-board air vehicle payload configuration includes a stabilized platform with relevant electro-optical sensor and a video recorder for image and data recording. As the mission requires, it is possible to install on the stabilized platform, alternatively, one of the following sensors: High Resolution TV Camera, FLIR or LLLTV. The stabilized Platform is used for stabilizing the installed sensors both in azimuth and elevation which involves de-coupling the optical axis with respect to the air vehicle attitude and isolation from to air vehicle vibration. Moreover it performs all the planning commands transmitted from the ground operator or from the video tracking system and measures the azimuth and elevation angles of the optical axis with respect to a reference frame fixed to the air vehicle. The MIRACH 26 is currently fitted with a stabilized platform manufactured by ALENIA DAAS.
The Command and Control Station is housed in a truck mounted shelter and is equipped with the relevant trolley mounted power generator.
A crew of three operators is necessary to operate the CCS : an air vehicle and antenna group operator, a payload operator and a Station Commander.
The air vehicle can be launched and recovered either under CCS or in a completely automatic mode. During flight, control of the air vehicle is possible in two distinct modes: autonomus control mode and remote control mode. In the autonomus control mode the operator can change both the sequence of preprogrammed flight legs and flight program, and resume the air vehicle remote control to the permitted phases. In the remote control mode the operator can change the roll/heading, the pitch/altitude and the throttle/speed.

Communication from the CCS to the air vehicle and vice-versa is realised by means of a J-band anti-jamming digital system.

The Antennas Group is mounted on a truck flatbed and is connected to the CCS by a cable. It can be located far from the CCS in order to increase System survivability. It allows both lift/orientation of the steerable antenna and air vehicle communication via a radio link.

The Mission Programming and Evaluation Station is composed of an equipped truck mounted shelter and a trolley mounted power generator. The station includes all equipment necessary for planning, developing and recording missions performed in automatic navigation mode. Mission flight path is subdivided into way points for each of which, geographical coordinates, air vehicle performances and payload commands are defined.

Evaluation of mission results can be performed either in real time, using the continuos flow of information received from the Command and Control Station, or in play back. The Evaluation and Programming Station is connected

to the other stations by telephone and radio links.

The Launch and Recovery Station includes the Launch Station and the Recovery SubStation. The Launch Station allows loading of the air vehicle on the launch ramp and connection of the power supply to the same vehicle. In this phase the system performs a pre-flight functional check of the air vehicle and loads the mission plan. After which, the system ignites the take-off booster and launches the air vehicle.
Once landed the recovery sub-station allows air vehicle retrieval.

The Maintenance and Refurbishment Subsystem allows operational refurbishment of the whole system up to second level maintenance.

3. Data Link & Video-tracker System
The Data Link guarantees a bidirectional RF time-shared link between air vehicle and Antennas Group while digital modulation of the transmission data permits a high degree of link reliability. Spread-spectrum techniques (frequency hopping) are implemented to secure the comunication channel. The functional components of the Data Link comprises on-board equipment (VTTC) and AG installed equipment (SRTT). Figure 3 shows the Data Link block diagram.

The Data Link serves as a means of data and image transmission from air vehicle to Ground Control Station (down-link) and command transmission from GCS to air vehicle (up-link).

The VTTC processes the video signal receveid from the electro-optic sensor in order to generate LOS pointing correction commands (Video-Tracking function) necessary to track the moving target. These commands consist of azimuth and elevation rates used to control the stabilized platform.
The video-tracking function implemented is of a "digital image correlation" type which facilitates target tracking even in extremely complex environments.

Video signals transmitted by RPV via the radio link are displayed on the Command and Control Station monitor. Target detection,recognition and identification is achieved by the operator using the video monitor and the images transmitted from the air vehicle.

When the target is identified the operator can activate the video tracking function by transmitting a video tracking command (via the up-link) to the on-board video tracker equipment.
At this stage it is possible to activate the Autotracking mode. In this mode the air vehicle maintains a circular path above the target at constant altitude and velocity. The circular trajectory's centre is the target present position while the radius is equal to the RPV altitude. The Autotracking process generates the commands for the Autopilot (Altitude, Roll and Ground Speed). The Altitude and Ground Speed commands have those values stored at instant of the Autotracking mode insertion while Roll command depends on Pan and Tilt values.

Before activating the video-tracker function the target should lie within a reference window located at the centre of the monitor. When the video-tracker function is activated the image in the reference window is memorized. Target tracking is maintained by scanning the search window for an image equivalent to the one memorized. (see figure 4: Video Tracking Function block diagram).

The video tracker has the capability to refresh the reference image with the actual image (update command) in order to guarantee target tracking, even in the presence of target shape variation due to a change in perspective.
The video tracker is able to maintain target lock with a variation of the pixel target dimensions, due to a change in FOV and air vehicle-target distance, as long as this variation does not exceed (in total) 80 % of the pixel search area.
The operator avails of a vernier to fine adjust platform pointing commands generated by video-tracker, in this way, initial pointing error or video-tracker drift can be nulled. The vernier commands are added to the video-tracker corrections.
The video tracker, comparing the actual image with the previously acquired image, computes the different target displacements in terms of pixels according to the horizontal and vertical axes referenced at the image center.

These displacements are converted to azimuth and elevation command rates used to correct the sensor line of sight.

4. Target Location
Target location (the final phase of target acquisition, after target detection, recognition and identification) allows the coordinates (latitude, longitude, altitude)

determination of a fixed or moving target.

The target location function is computed (see fig. 5) within the CCS using the following parameters:
- FOV provided by the electro-optical sensor
- Az, El provided by the stabilized platform
- Attitude and Heading (ϕ, θ, ψ) provided by on-board electronics
- RPV's Present Position and Barometric altitude provided by on board electronics
- Enemy mapped terrain provided by Command & Control Station data base.

The target's present position is obtained from the sum of the RPV's present position and the Target position relative to RPV position (Delta North, Delta East).
The referenced RPV altitude, relative to terrain, is calculated by means of the difference between the barometric pressure altitude and the terrain altitude.
The block diagram of figure 5 shows the input/output of the algorithm.

In this way, by means of a "Scartometry" algorithm, it is possible to calculate the target present position in the presence of rough terrain.
The "Scartometry" algorithm is based on a recursive search of the target's present position along the LOS vector. Starting from the RPV's present position and comparing the test points with the terrain altitude it is possible to obtain the target present position within a predefined error.

Target location algorithm accuracy depends on:
- designator position, screen characteristic and sensor's FOV (target position fixing on the monitor)
- RPV's present position accuracy (for Lat, Long, altitude calculation)
- on-board sensor accuracy (for ϕ, θ, ψ, Az, El calculation)
- horizontal and vertical quantization error of the mapped terrain (for the iteration algorithm calculating target altitude)
- terrain orography

In the paragraphs which follows target location error analysis is provided under the following assumptions:
- RPV's present position calculated by means of a GPS receiver code P altitude accuracy 32 m [95%] [2 dRMS], latitude and longitude accuracy 18 m [95%] [2 dRMS]
- target altitude accuracy (mapped terrain) 12m [1 σ]
- error absence during target designation on the monitor

5. Target Location Error Analysis Using Error Propagation Theory

5.1 Coordinate Systems

The following coordinate systems, or frames, are used to calculate the target's present position.
Local Level frame,
is a right-handed, orthonormal coordinate system with its origin located at the RPV's centre of gravity and its axes point true north, east, and down.
The x axis points north; the y axis points east and the z axis points down (see fig. 7).
RPV body frame,
is a right-handed, orthonormal coordinate system with its origin fixed at the RPV's centre of gravity. The coordinate axes are also fixed to the RPV and are defined by the RPV construction axes. The x axis points in the forward direction, along the RPV centerline; the y axis points to the right of the vehicle (along the right wing), and the positive z axis points towards the base of the vehicle (see fig. 7).
Line of Sight frame,
The Line of Sight frame is a right-handed, orthonormal coordinate system with its origin fixed at the sensor centre. The x axis points along the LOS from the RPV to the Target and is obtained by rotating the local level frame through an angle ($\psi + \sigma$) about the z axis followed by a rotation through an angle γ about the y axis (see figures 7 and 8).
In the present analysis the basis vectors of the orthonormal frames are indicated with x_{ij} (see fig. 6) where

x defines basis vectors of the orthonormal frames
i specifies the utilized frame
- i=0 local level frame
- i=3 body frame
- i=6 line of sight frame

j specifies the axis
- j=1 x axis
- j=2 y axis
- j=3 z axis

With the above notations the symbol R_{ij} indicates the components of the line-of-sight in the i frame, along the j axis.

5.2 Angular Rotation

The angular rotation between the above defined frames (see fig. 6) produces the following identity (Body to Line of Sight frame transformation matrix):

$$[\ \phi\ ]\cdot[\ \Theta\ ]\cdot[\ \sigma\ ]\cdot[\ \gamma\ ] = [\ Az\ ]\cdot[\ El\ ]\cdot[\ \phi_I\ ] \qquad (5.1)$$

Note that each element of the identity is a rotation matrix and the identity does not depend on the angle ψ.
Using the angles γ, σ, Θ, ϕ, the rotation matrix becomes:

$$\begin{bmatrix} x_{31} \\ x_{32} \\ x_{33} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} x_{61} \\ x_{62} \\ x_{63} \end{bmatrix}$$

where the A_{ij} elements are: (5.1.A)

$A_{11} = \cos\gamma\cos\Theta\cos\sigma + \sin\Theta\sin\gamma$
$A_{12} = -\cos\Theta\sin\gamma$
$A_{13} = \sin\gamma\cos\Theta\cos\sigma - \sin\Theta\cos\gamma$
$A_{21} = \cos\gamma(\sin\Theta\sin\phi\cos\sigma + \cos\phi\sin\sigma) + -\sin\gamma\sin\phi\cos\Theta$
$A_{22} = -\sin\Theta\sin\phi\sin\sigma + \cos\phi\cos\sigma$
$A_{23} = \sin\gamma(\sin\Theta\sin\phi\cos\sigma + \cos\phi\sin\sigma) + +\cos\gamma\sin\phi\cos\Theta$
$A_{31} = \cos\gamma(\sin\Theta\cos\phi\cos\sigma - \sin\phi\sin\sigma) + -\sin\gamma\cos\phi\cos\Theta$
$A_{32} = -\sin\Theta\cos\phi\sin\sigma - \sin\phi\cos\sigma$
$A_{33} = \sin\gamma(\sin\Theta\cos\phi\cos\sigma - \sin\phi\sin\sigma) + +\cos\gamma\cos\phi\cos\Theta$

Using the angles ϕ_I, El, Az the rotation matrix becomes:

$$\begin{bmatrix} x_{31} \\ x_{32} \\ x_{33} \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix} \begin{bmatrix} x_{61} \\ x_{62} \\ x_{63} \end{bmatrix}$$

where the B_{ij} elements are: (5.1.B)

$B_{11} = \cos Az\ \cos El$
$B_{12} = -\sin Az\ \cos\phi_I - \cos Az\ \sin El\ \sin\phi_I$
$B_{13} = -\sin Az\ \sin\phi_I + \cos Az\ \sin El\ \cos\phi_I$
$B_{21} = \sin Az\ \cos El$
$B_{22} = \cos Az\ \cos\phi_I - \sin Az\ \sin El\ \sin\phi_I$
$B_{23} = \cos Az\ \sin\phi_I + \sin Az\ \sin El\ \cos\phi_I$
$B_{31} = -\sin El$
$B_{32} = -\cos El\ \sin\phi_I$
$B_{33} = \cos El\ \cos\phi_I$

5.3 Sensor Line of Sight

The components of the LOS vector (see fig.8) with respect to the local level axes (x_{0j}) are:

$R_{01} = R\cos\gamma\cos\psi_t$
$R_{02} = R\cos\gamma\sin\psi_t$
$R_{03} = -R\sin\gamma$

where
$\psi_t = \psi + \sigma$
R_{01} is the LOS projection along the geographic North
R_{02} is the LOS projection along the East
R_{03} is the RPV altitude relative to the terrain

5.4 Analysis

5.4.1 Horizontal LOS Components Standard Deviation

The error analysis is based on the LOS vector variations δR_{0i}, due to $\delta\gamma$, $\delta\psi_t$ and δR variations.
The δR_{0i} components are:

$$\delta R_{01} = \delta R\cos\gamma\cos\psi_t - R\sin\gamma\cos\psi_t\ \delta\gamma + - R\cos\gamma\sin\psi_t\ \delta\psi_t$$
$$\delta R_{02} = \delta R\cos\gamma\sin\psi_t - R\sin\gamma\sin\psi_t\ \delta\gamma + + R\cos\gamma\cos\psi_t\ \delta\psi_t$$
$$\delta R_{03} = -\delta R\sin\gamma - R\cos\gamma\ \delta\gamma$$

After a rotation of an angle ψ_t about the z axis these variations become

$$\delta R_{71} = \delta R\cos\gamma - R\sin\gamma\ \delta\gamma \qquad (5.2)$$
$$\delta R_{72} = R\cos\gamma\ \delta\psi_t \qquad (5.3)$$
$$\delta R_{73} = -\delta R\sin\gamma - R\cos\gamma\ \delta\gamma \qquad (5.4)$$

where δR_{71}, δR_{72} and δR_{73} are the components of LOS variations in the x,y and z directions respectively. For clarity the components may be defined as:

$R_\lambda = R_{71}$ LOS projection in the vehicle-target direction
$R_\mu = R_{72}$ LOS projection perpendicular to R_λ
$h = R_{73}$ vehicle height over target

From equation (5.4)

$$\delta R = -\delta h/\sin\gamma - R\ (1/\tan\gamma)\ \delta\gamma$$

Substituting δR into (5.2) and $\psi_t = \psi + \sigma$ into (5.3):

$$\begin{bmatrix} \delta R_\lambda \\ \delta R_\mu \end{bmatrix} = \begin{bmatrix} S_{11} & S_{12} & S_{13} & S_{14} \\ S_{21} & S_{22} & S_{23} & S_{24} \end{bmatrix} \begin{bmatrix} \delta h \\ \delta\psi \\ \delta\gamma \\ \delta\sigma\cos\gamma \end{bmatrix}$$

where the elements of the sensitivity matrix S_{ij} are:

$$S_{11} = -1 / \tan\gamma$$
$$S_{12} = 0$$
$$S_{13} = -R / \sin\gamma$$
$$S_{14} = 0$$

$$S_{21} = 0$$
$$S_{22} = R \cos\gamma$$
$$S_{23} = 0$$
$$S_{24} = R$$

Assuming the vector elements [δh , $\delta\psi$, $\delta\gamma$, $\delta\sigma\cdot\cos\gamma$] as statistical independent gaussian variables and using statistical analysis, it is possible to obtain a covariance matrix for which the nondiagonal elements are equal to zero.
In this case the standard deviations are:

$$\sigma_{R\lambda} = \sqrt{S^2_{11}\ \sigma^2_h + S^2_{13}\ \sigma^2_\gamma} \qquad (5.5)$$

$$\sigma_{R\mu} = \sqrt{S^2_{22}\ \sigma^2_\psi + S^2_{24}\ \sigma^2_{\sigma\cdot\cos\gamma}} \qquad (5.6)$$

The variations $\delta\gamma$ and $\delta\sigma$ due to $\delta\Theta$, $\delta\phi$, δAz and δEl can be obtained using the previously defined identity (5.1).

$$\begin{bmatrix} \delta\gamma \\ \delta\sigma\cdot\cos\gamma \end{bmatrix} = \begin{bmatrix} Z_{11} & Z_{12} & Z_{13} & Z_{14} \\ Z_{21} & Z_{22} & Z_{23} & Z_{24} \end{bmatrix} \begin{bmatrix} \delta\Theta \\ \delta\phi \\ \delta Az \\ \delta El \end{bmatrix}$$

where the elements Z_{ij} of the sensitivity matrix are:

$$Z_{11} = \cos\sigma$$
$$Z_{12} = -\cos\Theta \sin\sigma$$
$$Z_{13} = -\sin\Theta \cos\phi \sin\sigma - \sin\phi \cos\sigma$$
$$Z_{14} = \cos\phi_t$$

$$Z_{21} = \sin\sigma \sin\gamma$$
$$Z_{22} = \cos\Theta \sin\gamma \cos\sigma - \sin\Theta \cos\gamma$$
$$Z_{23} = \sin\gamma\,(\sin\Theta \cos\phi \cos\sigma - \sin\phi \sin\sigma) + \cos\gamma \cos\phi \cos\Theta$$
$$Z_{24} = \sin\phi_t$$

Assuming the vector elements [$\delta\Theta$, $\delta\phi$, δAz, δEl] as statistical independent gaussian variables and using statistical analysis, it is possible to obtain a covariance matrix for which the nondiagonal elements are equal to zero.
In this case the standard deviations are:

$$\sigma_\gamma = \sqrt{Z^2_{11}\sigma^2\Theta + Z^2_{12}\sigma^2\phi + Z^2_{13}\sigma^2_{Az} + Z^2_{14}\sigma^2_{El}}$$

$$\sigma_{\sigma\cdot\cos\gamma} = \sqrt{Z^2_{21}\sigma^2\Theta + Z^2_{22}\sigma^2\phi + Z^2_{23}\sigma^2_{Az} + Z^2_{24}\sigma^2_{El}}$$

Assuming equal standard deviations
$\sigma_\Theta = \sigma_\phi = \sigma_{ATT}$ (RPV Attitude Accuracy)
$\sigma_{Az} = \sigma_{El} = \sigma_{PLT}$ (Pointing Angle Accuracy)

and considering
$Z^2_{11}+Z^2_{12} = 1 - \sin^2\Theta \sin^2\sigma \le 1$
$Z^2_{13}+Z^2_{14} = 1 - \sin^2\phi_t \sin^2 El \le 1$
$Z^2_{21}+Z^2_{22} = 1-(\sin\gamma \cos\sigma \sin\Theta + \cos\gamma \cos\Theta)^2 \le 1$
$Z^2_{23}+Z^2_{24} = 1 - \cos^2\phi_t \sin^2 El \le 1$

σ_γ and $\sigma_{\sigma\cdot\cos\gamma}$ reduced to:

$$\sigma_\gamma \le \sqrt{\sigma^2_{ATT} + \sigma^2_{PLT}} \qquad (5.7)$$

$$\sigma_{\sigma\cdot\cos\gamma} \le \sqrt{\sigma^2_{ATT} + \sigma^2_{PLT}} \qquad (5.8)$$

These equations permit the evaluation of the maximum values of σ_γ and $\sigma_{\sigma\cdot\cos\gamma}$ independent of RPV attitude and Platform pointing angles.

Substituting (5.7) and (5.8) into (5.5) and (5.6) respectivly it is possible to obtain the horizontal LOS components standard deviation:

$$\sigma_{R\lambda} \le \sqrt{S^2_{11}\ \sigma^2_h + S^2_{13}\ (\sigma^2_{ATT} + \sigma^2_{PLT})} \qquad (5.9)$$

$$\sigma_{R\mu} \le \sqrt{S^2_{22}\ \sigma^2_\psi + S^2_{24}\ (\sigma^2_{ATT} + \sigma^2_{PLT})} \qquad (5.10)$$

5.4.2 Target Present Position Standard Deviation

Target position is obtained by the sum of vehicle present position and relative vehicle-target position. The target present position standard deviations are

$$\sigma_x = \sqrt{\sigma^2_{xRPV} + \sigma^2_{R\lambda}} \qquad (5.11)$$

$$\sigma_y = \sqrt{\sigma^2_{yRPV} + \sigma^2_{R\mu}} \qquad (5.12)$$

where σ_{xRPV} and σ_{yRPV} are the vehicle present position standard deviations.

6. Results

Utilizing (5.9), (5.10), (5.11) and (5.12) is possible to calculate the sensors' precision.
Using the following data

CEP	= 50 m
Height	= 1500 m
Slant Range	= 2500 m

$$\sigma_h = \sqrt{\sigma^2_{hGPS} + \sigma^2_{hMAP}} = 20 \text{ m}$$

$$\sigma_{xRPV} = \sigma_{yRPV} = \sigma_{RPV} = 9 \text{ m}$$

is possible to study the correlation between σ_ψ σ_{ATT} and σ_{PLT}.
Fig. 9 shows σ_{ATT} vs σ_{PLT} for different values of σ_ψ.
Assuming the following accuracies (1 σ) for the air vehicle attitude and heading angles:

$\sigma_\psi = 14$ [mrad]
$\sigma_\phi = \sigma_\Theta = 7$ [mrad]

the platform accuracies become

$\sigma_{Az} = \sigma_{El} = 6$ [mrad]

From (5.7) and (5.8) the following values of σ_γ and $\sigma_{\sigma \cdot \cos\gamma}$ are obtained

$\sigma\gamma = \sigma_{\sigma \cdot \cos\gamma} = 9$ [mrad]

Figg. 10 and 11 show the target CEP variation under RPV present position (lat, lon, height) variation; it is possible to evalutate the influence of the RPV present position degradation on the target position precision.
Fig. 10 shows $\sigma_\gamma = \sigma_{\sigma \cdot \cos\gamma}$ vs σ_ψ for different values of CEP.
Target CEP vs air vehicle CEP for different altitude standard deviation σ_h values are shown in Fig. 11.

Using the Monte Carlo Method with the same input values as above the Probabilty Distribution was obtained as a function of Radial Error.

Where $\text{Rad Err} = \sqrt{err^2(\lambda) + err^2(\mu)}$
and $err(\lambda)$ and $err(\mu)$ are the errors on the ground along the target local axes (see Fig. 12). This Method estimated a CEP of 47 meters. The comparison of the results obtained through the two methods is satisfactory in so far as there are no substantial differences.

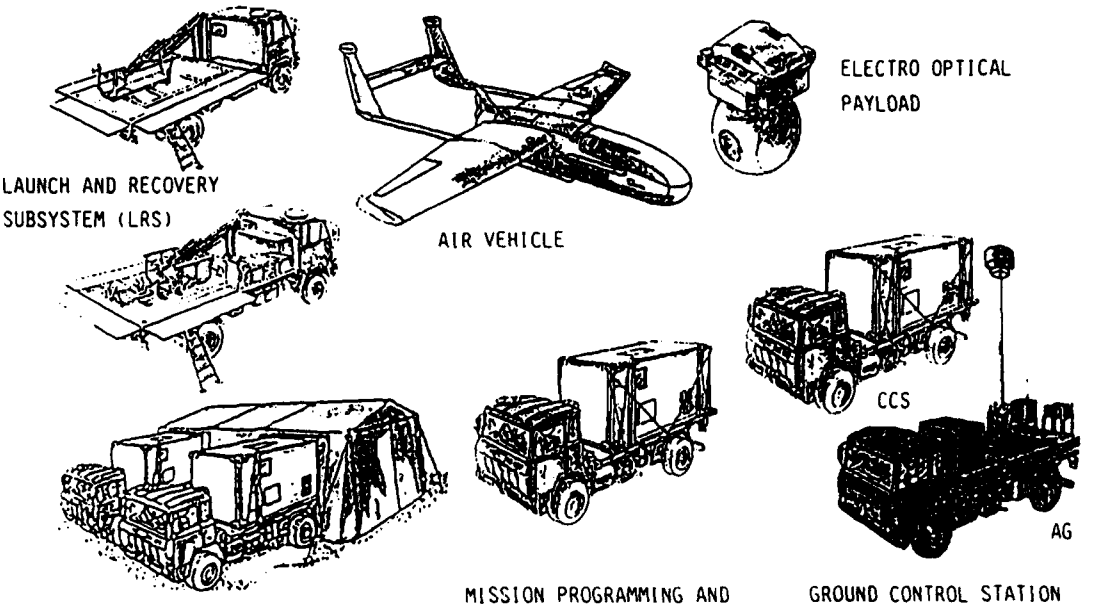
7. Conclusions

The results obtained from this analysis indicate that such a target position accuracy (50 m CEP under conditions defined in para 6) is obtainable using commercial onboard sensing technology (AHRS, GPS and Platform) and digital mapped terrain.

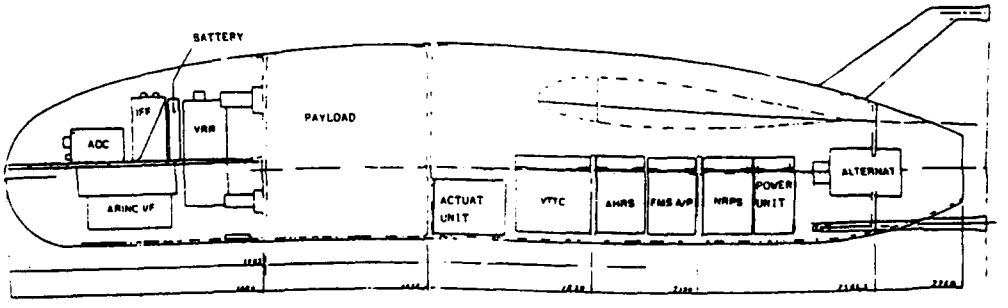
As regards target acquisition missions, automated target search software is under development to maximize target acquisition probability hence reducing the operator workload. The system provides a ground program phase to define the flight path and area, line or point search routines to perform the reconnaissance. Upon definition of the search method, the system defines the best UAV flight path to obtain: a minimum flight time over the interested zone, a total area acquisition as a function of the optical sensor footprint and sufficient image recognition time.

8. References

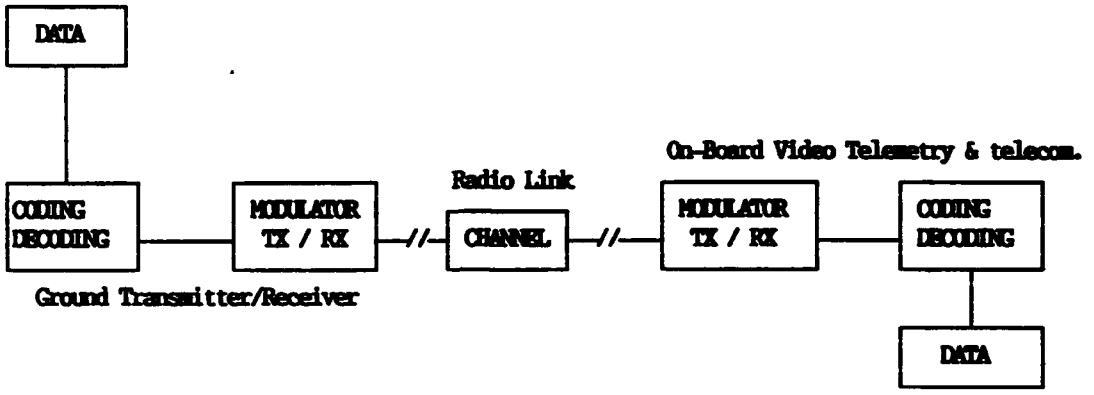
[1] M. Kayton and W. R. Fried, "Avionics Navigation Systems", John Wiley & Sons, Inc., New York, 1969



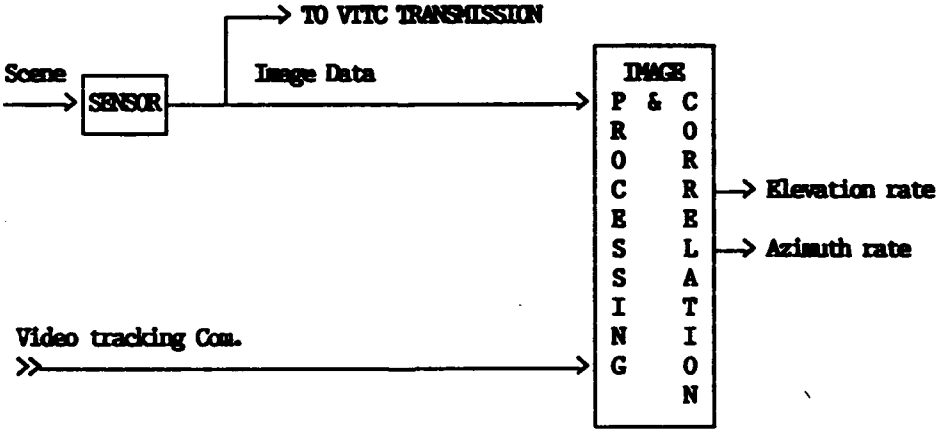
- Figure 1 : MINI RPV SYSTEM -



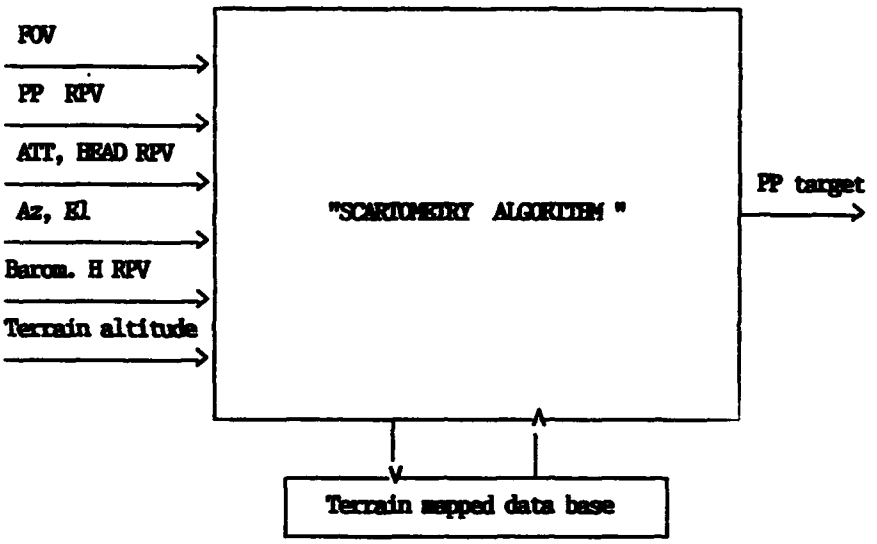
- Figure 2 : MIRACH 26 ON BOARD ELECTRONICS -



- Figure 3 : Ground transmitter/Receiver - On board Video Telemetry and Telecommand Block Diagram -

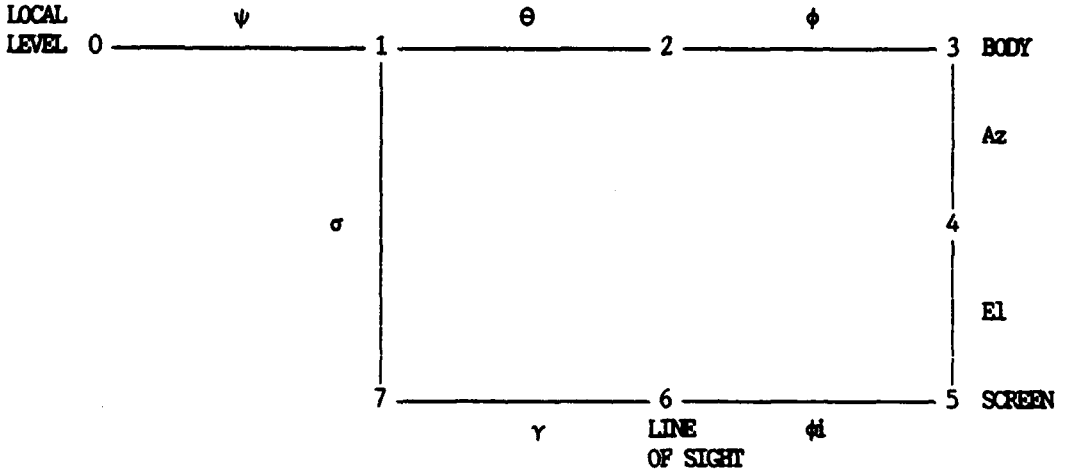


- Figure 4: Video tracking function block diagram -

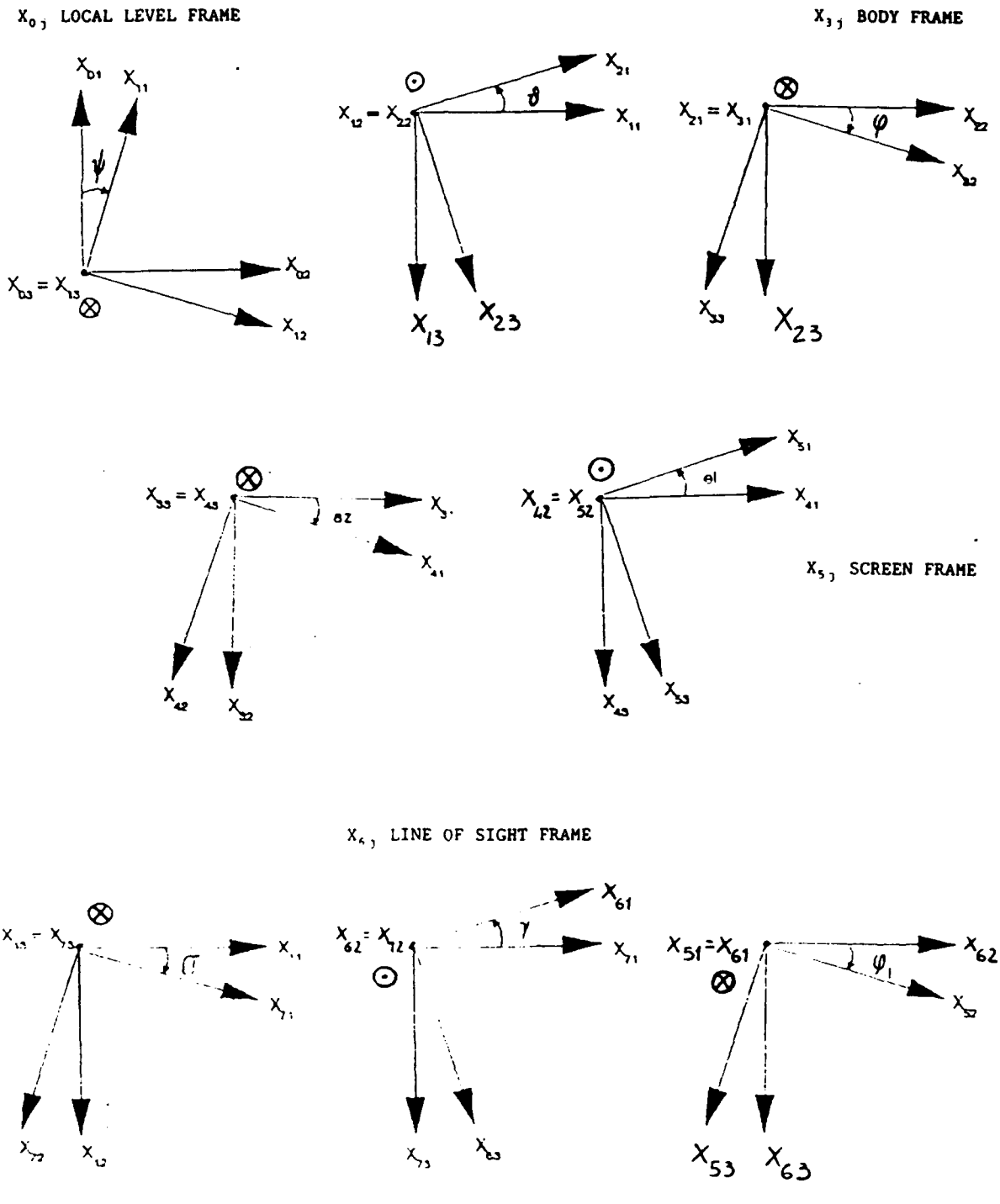


FOV	Field of view (opt. sensor)	Az, El	Azimuth, Elevation (Stab. platform)
PP RPV	RPV present position	Barom. H RPV	Barometric altitude (RPV)
ATT, HEAD RPV	Attitude & Heading (RPV)	PP target	Target present position

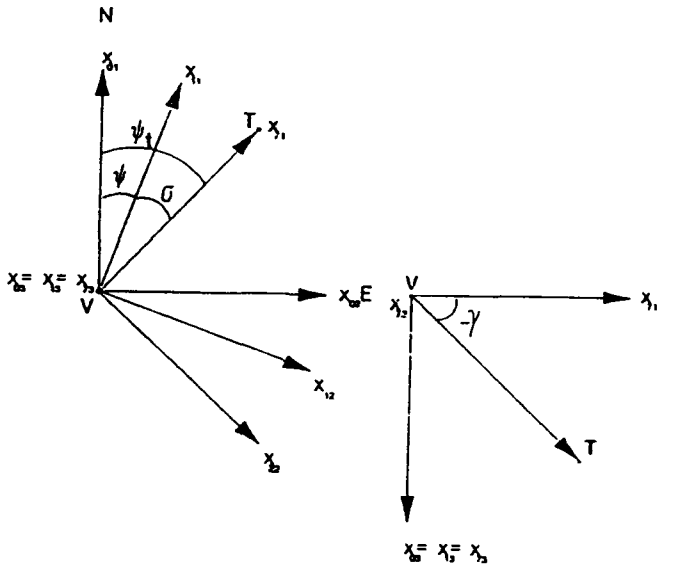
- Figure 5 : Target Location Function (in Command & Control Station) -



- Figure 6: Reference Frames Angular Relations -

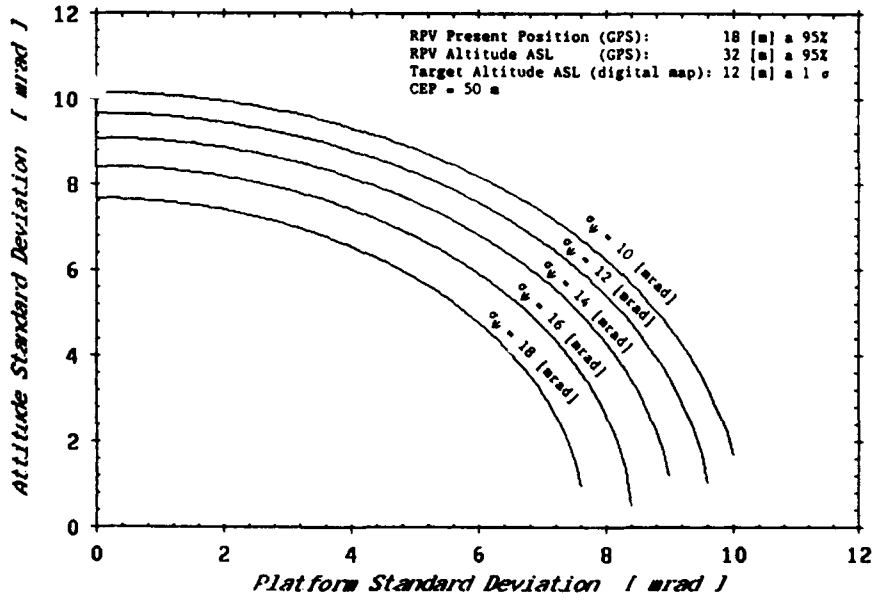


- Figure 7: Reference Frames Angular Rotation -

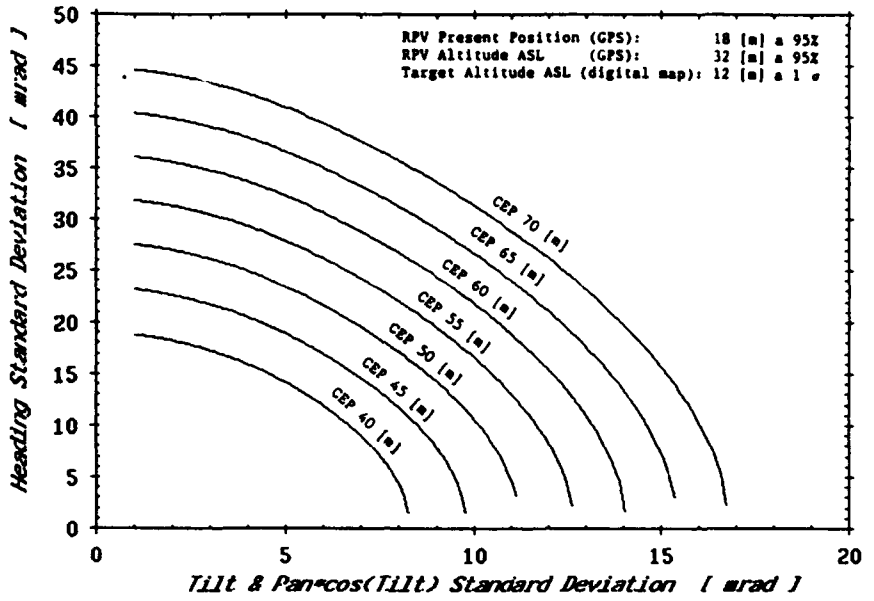


V = Vehicle
T = Target

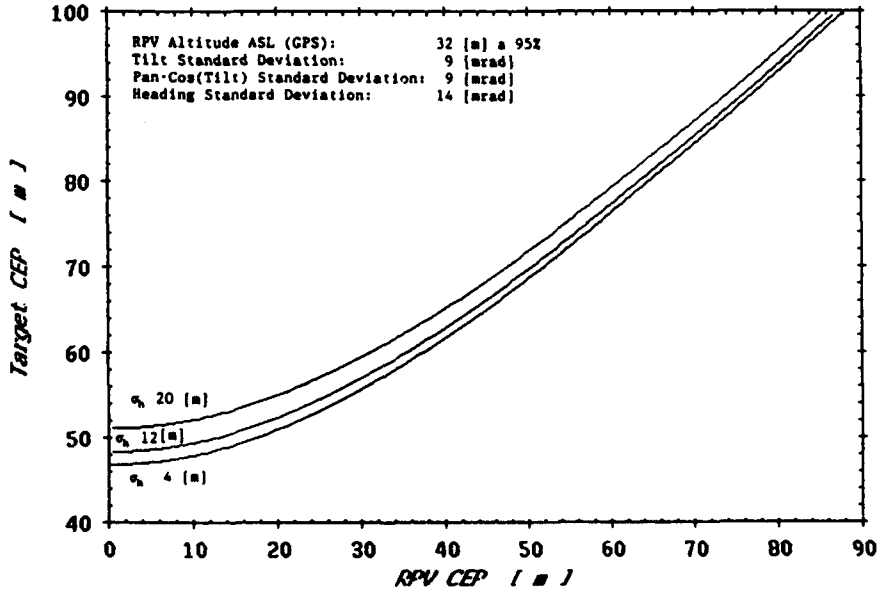
– Figure 8: LOS vector components –



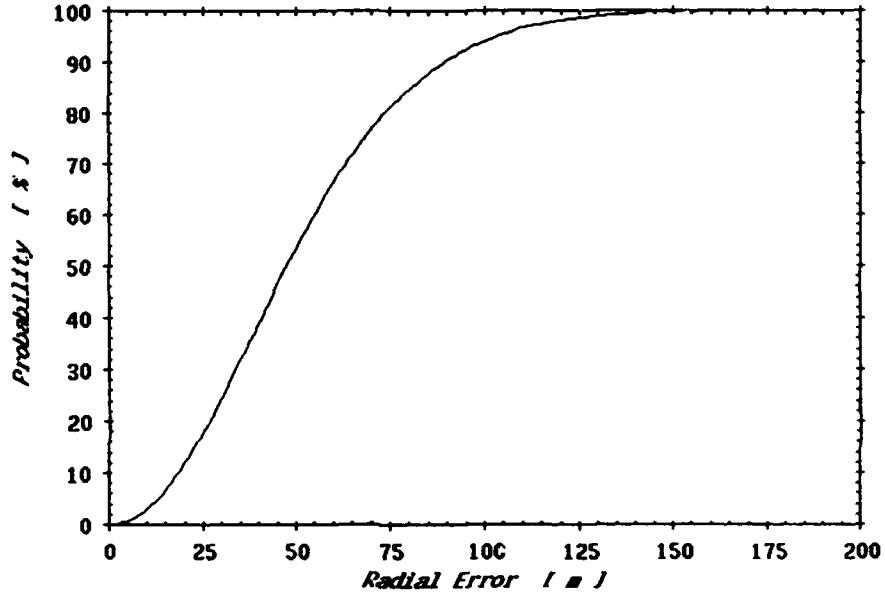
– Figure 9 : σ_{ATT} vs σ_{PLT} for different σ_{ψ} –



- Figure 10 : σ_ψ vs σ_γ and $\sigma_{\sigma \cdot \cos\gamma}$ -



- Figure 11 : Target CEP Variation vs Vehicle CEP Variation -



- Figure 12: Radial Distribution -

ETABLISSEMENT DE LA SITUATION AERIENNE DANS UN ENVIRONNEMENT MULTISENSEUR MOBILE

C. RIVIERRE
M. DESBOIS

THOMSON-CSF
7, rue des Mathurins
92220 BAGNEUX
FRANCE

PREFACE

Cette communication traite de l'impact du Concept de mobilité sur un système de commandement et de contrôle de l'espace aérien. Le sujet se limite à l'établissement de la situation aérienne. Les différents aspects nouveaux liés à la mobilité sont abordés ainsi que les moyens de résoudre cette difficulté supplémentaire au niveau de la fusion des données. Ces moyens reposent sur une expérience originale, sur des résultats d'expérimentation et sur des travaux d'étude. De futures évolutions du système sont également envisagées à la fin (mettant plus à contribution les éléments aéroportés) du papier.

TABLE DES MATIERES

RESUME

La maitrise de la situation aérienne est un aspect fondamental du théatre des opérations dans un conflit moderne. La plupart des systèmes de commandement et de contrôle, plus connus sous le sigle C2, ont été conçus pour mener des opérations sur le territoire national, ou du moins dans la zone d'intérêts et de responsabilité d'un pays.

L'augmentation de la portée des armements, ainsi que la diversification de la menace ont contribué à accroitre considérablement le volume dans lequel la maitrise de la situation reste une priorité. Parallèlement, l'évolution de la menace, notamment dans le domaine des missiles à long rayon d'action (missiles balistiques ou de croisière), porte à envisager le déplacement des centres de commandement et de contrôle sur des points avancés, des territoires alliés, et/ou au fur et à mesure du déplacement des zones de conflit, loin des installations fixes et des moyens de détections sédentaires.

Ces données bousculent considérablement l'architecture conventionnelle des systèmes. Dans cet environnement nouveau, des centres mobiles de commandement et de contrôle devront rassembler les informations de diverses sources de détection, mobiles ou déplaçables elles aussi, en une représentation précise et compréhensive de la situation aérienne. Cette mobilité, tout autant que la déformation du réseau de senseurs, complexifie les problèmes déjà nombreux que doit résoudre la poursuite.

L'exposé rappelle que la poursuite regroupe les moyens de collecte, de fusion, de filtrage et de synthèse des informations délivrées par les senseurs dans le but d'établir la situation aérienne. Ces informations, obtenues avec la coopération de la cible ou non, résultent de mesures actives et/ou passives ; elles peuvent contenir des éléments d'identification ou de classification des objets détectés. C'est grâce à une connaissance précise de la situation aérienne renseignée (RAP : Recognized Air Picture) que les différentes possibilités de manoeuvres des véhicules aériens pourront être gérées, contrôlées et analysées. Des techniques d'évaluation de la situation pourront alors être utilisées à partir de cette image.

Outre les problèmes de transmission des informations, qui peuvent être traités par des moyens de communication fiables et efficaces tels que les liaisons tactiques ou les systèmes de distribution des informations protégées (JTIDS : Join Tactical Information Distribution System) la gestion des informations provenant d'un réseau évolutif pour le problème du maintien de la cohérence des données.

Le sujet développé dans cet exposé traite d'une part de systèmes opérationnels composés de senseurs fixes et répartis et des techniques mises en oeuvre pour fusionner les données de ces senseurs, d'autre part d'études sur les systèmes mobiles aéroportés existants ou en projet. Ces études de senseurs colocalisés sur système mobile permettent d'envisager des solutions à base de senseurs fixes et/ou mobiles répartis.

Les performances sont améliorées et les délais de réaction réduits par le réhaussement de la probabilité de détection multisenseur, la vulnérabilité du système est abaissée et la résistance au brouillage accrue grâce à la prise en compte des mesures multidimensionnelles.

Les bénéfices d'un tel système multisenseur perfectionné peuvent être ruinés par

l'imprécision de localisation et de positionnement des divers senseurs qui participent à l'élaboration de la situation aérienne. L'effort particulier nécessaire à l'estimation continue des biais des senseurs et également à la surveillance de la qualité des senseurs est particulièrement traité dans cet exposé.

L'optimisation des moyens de détection disponibles permet à la fois d'apporter une meilleure sécurité aux senseurs en leur imposant des modes passifs plus discrets ou en commandant des contre-mesures, et d'affecter au mieux la capacité globale de détection. Des résultats d'investigations expérimentales seront présentés, et plus particulièrement la visualisation des domaines de couverture multisenseur regroupés dans des images synthétiques permettant aux opérateurs du système de connaître à chaque instant les possibilités de détection du système en tenant compte de l'environnement de chaque senseur et des missions du système.

En conclusion, l'exposé montre que les investigations menées et les leçons apprises permettent non seulement de définir l'architecture des futurs systèmes multisenseur mobiles/déplaçables, mais également de prévoir quelles seront leurs performances et leurs capacités.

MOTS-CLES

Poursuite Multisenseur MST - Fusion de données - MP-VU - Gestion Multisenseur.

SOMMAIRE

L'établissement de la situation aérienne est un aspect fondamental du théatre des opérations dans un conflit moderne. De plus, l'augmentation de la portée des armements ainsi que la diversification de la menace contribuent à accroître considérablement le volume dans lequel la maîtrise de la situation reste une priorité. Ceci porte à envisager le déplacement des centres de commandement et de contrôle sur des points avancés, des territoires alliés, et/ou au fur et à mesure des zones de conflits, loin des installations fixes et des moyens de détections sédentaires.

L'exposé rappelle que l'établissement de la situation aérienne regroupe les moyens de collecte, de fusion, de filtrage et de synthèse des informations délivrées par les senseurs, afin de gérer, contrôler et analyser les différentes possibilités de manoeuvre des véhicules aériens.

La mobilité du dispositif de surveillance, tout autant que les déformations du réseau de senseurs qui le composent rendent difficile la réalisation de cette mission :

- Il faut connaître les variations des recouvrements de couverture et s'y adapter.
- Il faut rapidement corriger les erreurs de calage entre les senseurs.
- Il faut pouvoir fonctionner en environnement de fort brouillage.

Les réalisations et investigations menées par THOMSON-CSF sur le sujet permettent non seulement de définir l'architecture des futurs systèmes multisenseurs mobiles/déplaçables mais également de prévoir quelles seront leurs performances et leurs capacités.

1. INTRODUCTION

La maîtrise de la situation aérienne est un aspect fondamental du théâtre des opérations dans un conflit moderne. Une illustration de l'importance du contrôle des opérations aériennes nous a, par exemple, été récemment donnée lors de la guerre du Golfe.

Cette maîtrise est essentielle, non seulement pour mener à bien des actions offensives, mais également pour assurer la mission de défense d'un territoire.

La plupart des systèmes de commandement et de contrôle, plus connus sous le sigle C^2, ont été conçus pour mener des opérations sur le territoire national, ou du moins, dans la zone d'intérêt et de responsabilité d'un pays. Le déplacement de la menace de conflits Est-Ouest en tensions Nord-Sud, ainsi que la mondialisation des conflits (on parle de "Nouvel Ordre Mondial") et l'extension des besoins de défense du sanctuaire national mais aussi des intérêts économiques en dehors des frontières, conduisent à envisager le déplacement des centres de commandement et de contrôle, (ou de parties importantes de leur structure) :

- sur des points avancés,
- sur des territoires alliés,
- au fur et à mesure du déplacement des zones de conflits,
- loin des installations fixes et des moyens sédentaires de détection et de communication.

L'augmentation de la portée des armements (missiles balistiques, missiles de croisière, capacité de ravitaillement en vol des chasseurs), la diversification des moyens mis en oeuvre ainsi que la généralisation des systèmes de commandement et de coordination inter-armées contribuent à accroître considérablement le volume dans lequel la maîtrise de la situation reste une priorité.

2. EXPOSE DU PROBLEME

Dans cet environnement nouveau, des centres mobiles de commandement et de contrôle devront rassembler les informations des diverses sources de détection, mobiles ou déplaçables elles aussi, dans le but de produire une représentation précise, fiable et compréhensive de la situation aérienne. Cette situation renseignée pourra ensuite être distribuée aux différents utilisateurs et permettre notamment le guidage et le pilotage des engins aériens en mission. Ainsi, la surveillance est à considérer comme un "service" à la disposition des autres fonctions d'un centre C^2.

Les principales missions de la fonction de surveillance sont :

- la production et l'entretien de la situation aérienne renseignée,
- la gestion des moyens de surveillance,

- la distribution de la situation aérienne,

- les échanges avec les autres systèmes (inter-armée).

La fonction de surveillance regroupe les moyens de collecte, de fusion, d'identification et de synthèse des informations délivrées par les senseurs (et autres systèmes de détection) dans le but d'établir la situation aérienne.

Les composantes de la surveillance sont donc :

- les senseurs et moyens de détection rattachés,

- les moyens de communication,

- les centres de fusion des données.

Les informations sur les véhicules aériens sont obtenues avec ou sans la coopération des cibles, et résultent de mesures actives ou passives. Ces mesures peuvent également contenir des éléments d'identification ou de classification des objets détectés.

La mobilité du dispositif de surveillance, tout autant que les déformations du réseau de senseurs qui le composent, complexifient les problèmes déjà nombreux que la poursuite doit résoudre.

Les principales difficultés liées à la mobilité sont les suivantes :

- Impact au niveau des senseurs : les senseurs mobiles (ou déplaçables) possèdent des caractéristiques moindres que leur version fixe (puissance d'émission et taille de l'antenne réduite, contraintes liées à la mobilité, renforcement des équipements de structure, durcissement des cartes électroniques).

- Impact sur la localisation des senseurs : la mobilité introduit une incertitude sur la position géographique, sur la verticalité et sur le calage au nord des senseurs. Des moyens grossiers de positionnement et de calage seront généralement utilisés lorsque le temps d'installation est réduit et si la zone de déploiement n'autorise pas le fonctionnement optimal des moyens de localisation par satellite.

- Impact sur la transmission des données : le volume des données, ainsi que la qualité et/ou la fiabilité des informations transmises par les senseurs sont directement corrélés au type des liaisons utilisées.

- Impact sur l'homogénéité de la situation aérienne : le réseau de senseurs et autres moyens de détection étant évolutif, la fusion doit être fortement adaptative à ces changements.

3. EXPERIENCE THOMSON-CSF

THOMSON-CSF a réalisé une poursuite multisenseur basée sur une technique de fusion de plots avec mise à jour des pistes à intervalles variables (MST/MP-VU). Cette technique offre de nombreux avantages naturels, auxquels s'ajoutent des choix industriels de conception et d'implémentation qui permettent de résoudre la majorité des problèmes énoncés dans la première partie de cet exposé.

Les problèmes à résoudre sont :

- s'adapter aux variations des recouvrements de couverture et des trous de détection,

- corriger rapidement les erreurs de calage entre les senseurs,

- accepter des mesures de senseurs variés (précision, cadence des mesures, modes de fonctionnement, informations mesurées, etc),

- supporter les changements de type de senseur et du nombre de senseurs,

- pouvoir représenter la couverture courante du réseau de senseurs,

- fonctionner en environnement de fort brouillage.

Les sections suivantes explicitent la nature de ces problèmes et les moyens de s'en affranchir.

3.1. Calage des senseurs

Une condition préalable à un pistage multisenseur efficace est la résolution des problèmes d'homogénéité. Toute implémentation de pistage multisenseur avec une estimation médiocre des biais des senseurs provoquera l'augmentation de la taille des fenêtres de corrélation, compromettant ainsi le bénéfice de leur réduction obtenue grâce aux intervalles de temps courts entre les mises à jour. De plus, la prise en compte d'informations mutisenseur dans de mauvaises conditions de calage provoquerait des fausses corrélations.

Les erreurs sont provoquées par des bruits systématiques qui diffèrent pour chaque senseur (macro erreurs). Ces "macro erreurs" sont les erreurs de positionnement des radars, les biais en site, azimut ou distance, les erreurs de datation des informations mesurées, les erreurs de squint ou de verticalité du faisceau d'antenne. De tels biais sont critiques dans une poursuite multisenseur car ils provoquent des oscillations de pistes, accentuées par le faible espacement des mises à jour et sont perçues comme de fortes accélérations des pistes entre les mises à jour.

Les "macro erreurs" peuvent changer brusquement, ce qui peut être dû à des maintenances techniques sur les senseurs, à l'influence que le vent a sur la mécanique de l'antenne du senseur ou à un mauvais repositionnement d'un senseur mobile. Ceci impose une estimation dynamique de ces "macro erreurs" en utilisant la mise à jour des pistes.

C'est pourquoi une fonction d'estimation des erreurs systématiques calcule, en continu, les erreurs systématiques par senseur, dans un environnement multisenseur ayant un recouvrement suffisant. Les valeurs calculées sont :

- les erreurs de position des senseurs dues à un mauvais positionnement,

- les biais en site dus à un mauvais réglage du site de l'antenne,

- les biais en azimut dus à un mauvais calage au nord de l'antenne,

- les biais en distance mesurée dus à un mauvais réglage du kilomètre zéro de l'antenne,

- les biais de datation dus aux erreurs systématiques de datation des plots (temps de stockage, temps de transmission),

- les erreurs de déformation du faisceau (squint) dues à l'extracteur pour des radars ayant un diagramme d'antenne dont le site varie selon l'azimut,

- les erreurs de verticalité dues à la non verticalité de l'axe de rotation du senseur.

Les biais de position et de mesure des différents senseurs sont traités comme le vecteur d'état d'un estimateur et sont mis à jour chaque fois que le senseur fournit une nouvelle mesure qui corrèle avec une des pistes sélectionnée. Pour optimiser le choix des données utilisées par le filtre, une sélection des pistes utiles est faite pour ce filtre. Ces pistes sont sélectionnées selon des critères tels que le nombre de senseurs qui détectent la piste et la cinématique de cette piste.

3.2. Résistance au brouillage

Les systèmes opérationnels basés sur la technique MST/MP-VU incluent des capacités de traitement des strobes. Les plots peuvent avoir 2 ou 3 dimensions selon la capacité du radar, mais il existe aussi des données à 1 ou 2 dimensions (strobes) fournies par un radar (mode ECM), par un senseur ESM ou par un senseur PJL, lorsqu'une cible commence à brouiller ou émettre un signal radio-électrique.

THOMSON-CSF, grâce à son expérience, est partisan d'une architecture multisenseur comprenant la fusion directe des données provenant des senseurs actifs ou passifs. Ceci est fondé sur :

- une corrélation directe des strobes avec les pistes existantes,

- une création automatique de piste avec une fonction de recherche d'intersection vraisemblable (deghosting) parmi les strobes restants (avec l'aide d'un opérateur pour cette fonction),

- un processus de mise à jour utilisant aussi bien les plots que les strobes.

Actuellement, c'est la meilleure approche satisfaisant les exigences suivantes :

- traitement des données passives avec la capacité de faire face à un déploiement réduit de senseurs,

- résistance contre le brouillage intermittent,

- utilisation du meilleur parti de chaque information,

- charge de travail minimisée de l'opérateur pour le processus de fusion,

- transition aisée du pistage actif au pistage passif.

Tout ceci est possible grâce à la capacité de mettre à jour une piste en utilisant un filtre comprenant une boucle de partage des mesures, c'est à dire que contrairement à un filtre (ou banc de filtres) conventionnel, des mesures partielles d'azimut de distance ou de site peuvent être introduites dans le filtre.

Ainsi THOMSON-CSF a rendu possible la mise à jour automatique de pistes multisenseur avec :

- uniquement des plots "normaux",

- des plots "normaux" et des plots strobes,

- uniquement des plots strobes.

3.3. Gestion des senseurs

L'augmentation de la variété et des performances de la menace exige l'adaptation du système aux contraintes du théatre des opérations et nécessite l'exploitation optimale de la diversité des senseurs. Ceci ne peut être fait manuellement, il est donc nécessaire d'utiliser une fonction automatique de gestion multisenseur.

Cette fonction opère sur des directives opérationnelles comme :

- améliorer la surveillance dans des zones particulières,

- opérer en mode discret tout en conservant une qualité satisfaisante à la situation aérienne,

- augmenter la capacité de détection pour des cibles de faible surface équivalente dans certains couloirs de pénétration,

- suivre un plan de fréquence particulier (ce qui peut conduire à n'utiliser que certains senseurs ou à émettre en mode sectorisé).

Cependant, des demandes très prioritaires peuvent être exprimées à ce niveau :

- arrêt de l'émission ou émission à faible puissance pour un radar (en cas d'attaque d'ARM (Anti Radar Missile) par exemple),

- avoir une cadence de mise à jour plus élevée pour certaines cibles désignées.

Pour remplir ce rôle, cette fonction doit évaluer la couverture multisenseur. L'évaluation opère à partir d'informations fournies par différentes sources telles que les données de configuration géographique du système (terrain, positionnement des senseurs) ou les données d'environnement fournies par les senseurs ou introduites par l'opérateur, tout ceci dans le but d'évaluer la qualité de la couverture multisenseur du système et sa performance (en temps réel ou en mode de prédiction).

3.4. Avantages de la poursuite MST/MP-VU

La poursuite MP-VU industrialisée par THOMSON-CSF offre notamment les avantages suivants :

- exploitation de la redondance de couverture par l'utilisation de toutes les mesures au fur et à mesure de leur détection,

- amélioration des performances, notamment réduction des durées d'initialisation, détection rapide des évolutions des véhicules aériens et précision accrue pendant leurs manoeuvres,

- mise à profit du réhaussement de la probabilité de détection par "l'effet multisenseur",

- souplesse de la poursuite vis-à-vis de la

diversité des informations à fusionner (plots 3D, 2D, vitesse radiale, cadences et précisions des mesures différentes),

- résistance au brouillage grâce à la possibilité de traitement des mesures partielles (mise à jour des pistes avec les strobes, traitement des intersections de strobes multisenseurs),

- surveillance automatique des senseurs, évaluation et correction des biais des senseurs concurremment à l'élaboration de la situation aérienne.

Le niveau de performance atteint est conforté par les résultats obtenus sur les sites réels où THOMSON-CSF a installé la technique MST/MP-VU. En s'appuyant d'une part sur cette expérience opérationnelle et d'autre part sur les études en cours, THOMSON-CSF envisage des solutions à base de senseurs fixes et/ou mobiles répartis.

4. EVOLUTIONS POSSIBLES

Les véhicules aériens sont à la fois les acteurs et les moyens de la surveillance. Lorsque des moyens de communication sûrs et performants seront généralisés à bord des aéronefs, non seulement les véhicules aériens amis pourront participer à l'élaboration de la situation aérienne par leurs propres détections, rendues exploitables grâce à leur précision de localisation et à la richesse des informations transmises (Cf. catalogue de messages MIDS), mais ils pourront recevoir une partie de la situation élaborée au sol. Ainsi, leur situation locale sera enrichie. De plus, le profil de leur mission pourra être modifié par la possibilité de mettre en oeuvre leurs senseurs actifs uniquement en phase terminale.

5. CONCLUSIONS

Il résulte que lorsque l'on compare les avantages que procure une telle implémentation de la fonction de surveillance avec les problèmes spécifiques posés par la mobilité des systèmes de surveillance de l'espace aérien, on obtient le tableau suivant qui souligne l'adéquation de la solution préconisée au problème posé :

Senseurs moins performants
∧
|
∨

- Exploitation de "l'effet multisenseur", inté gration de senseurs variés et notamment aéroportés (plus rapprochés)

Localisation peu précise
∧
|
∨

- Estimation automatique des biais

- Combinaison avec les systèmes de communication/ localisation aéroportés

Variation du réseau de senseurs (en nombre et en géométrie)
∧
|
∨

- MP-VU = monosenseur à chaque instant
- Optimisation du réseau de détection

REFERENCES

- M.Carpentier (1984), "Radars, Bases Modernes", Masson.
- E.Roubine, J-Ch.Bolomey, S.Drabovitch and C.Ancona (1978), "Antennes", Masson.
- F. Le Chevalier (1989), "Principes de Traitement des Signaux Radar", Masson.
- A.Farina and F.A.Studer (1986), "Radar Data Processing", Research Studies Press LTD.
- Dr J.Llinas and D.Hall (1989), "Data Fusion", State of the Art LTD.
- S.S.Blackman (1986), "Multiple-Target Tracking with Radar Application", Artech House INC.
- M.Desbois (1987), "Multiple Radar Tracking and real time processing", Milcomp'87 in London U.K., sep 87.
- M.Desbois (1988), "Multisensor Tracking", Signal Magazine, Nov 88.
- M.Desbois (1990), "Detection means management in a multisensor environment", Radarcon'90 in Adelaide Australia, Apr 90.

LISTE DES SYMBOLES

ARM	Anti-Radiation Missile
C^2	Command & Control
ECM	Electronic Counter-Measure
ESM	Electronic Support Measurement
MIDS	Multifunction Information Distribution System
MP-VU	Multiple Plot-Variable Update
MST	MultiSensor Tracking
PJL	Passive Jammer Locator.

AIR SITUATION ESTABLISHMENT IN A MOBILE MULTISENSOR ENVIRONMENT

C. RIVIERRE
M. DESBOIS

THOMSON-CSF
7, rue des Mathurins
92220 BAGNEUX
FRANCE

PREFACE

This paper applies to the impact of the mobility concept on a Air Comand & control system. The subject is limited to the air picture establishment. Various aspects linked with mobility are pointed out as well as solutions to handle the additional difficulty at the data fusion level. These solutions are based on an original experience, on experiment resultts, and on study work. Advanced system capabilities which make more intensive use of airborne platforms, are also discussed at the end of the paper.

TABLE OF CONTENTS

RESUME

Keeping control of the air situation is a fundamental element in managing the theatre of operations in modern conflicts. Most of the command and control systems, well known under the C2 acronyn, have been designed to carry on operations on national territory, or at least in a country's area of interest or responsibility.

The increase in the arms range of efficiency as well as in the variety of the threat enlarge the volume in which having air control is a top priority. Concurrently, the development of weapons, mainly long range missiles (cruise missiles and balistic missiles), leads us to envisage the possibility of moving forvard the command and control centres to advanced posts or allied countries, or of having then follov the conflict area, far from fixed groundbased facilities and sedentary detection means; all this contributes to rethinking of conventional system designs.

In this nev environment, mobile command and control centres will have to gather sensor data from removeable and mobile detection means, and to produce an accurate and understandable air situation. Sensor mobility and dramatic shape changes in sensor networks both make fusion process issues more complicated.

This paper points out that multisensor tracking includes sensor data collection, plot fusion, smoothing and synthesis of measures delivered by the sensor network, in order to assess the air picture. Target information is obtained with or without cooperation of the aircraft, and results from active or passive measures; it can include identification data, and classification of the target. A precise RAP (Recognised Air Picture) allovs support of air vehicle mission control and management. Then, situation evaluation techniques can be applied on such an accurate air picture.

As well as information exchange, which can be supported by the means of reliable and efficient systems such as tactical links or joint tactical information distribution systems, the management of data from a moving sensor network is complicated by the issue of preserving multisensor data consistency.

This paper will points out on one hand operationel systems based on fixed, groundbased and distributed sensors, with associated techniques perform the fusion of data provided by such a sensor network. On the other hand, studies about mobile airborne/on board existing or planned systems will be shown: these systems are mainly built out of colocated sensors. Lessons learned from both types of systems allov us to investigate solutions based on fixed and mobiles distributed sensors.

Raising multisensor target detection probability improves tracking performance and reduces reaction time, vhile multidimentional measurement processing reduces system vulnerability and enhances resistance to jamming.

All ensuing benefits of such an improved multisensor system might be shattered by inaccuracies in the positionning and alignment of the sensors which contribute to the air picture assessment. This paper will pinpoint the specific effort required for on-line sensor bias estimation and for sensor data quality supervision.

Optimisation of available detection means ensure both sensor safety (by forcing more discreet detection modes and electronic counter-counter measures) and global surveillance distribution to allocated sensors. Results from experimental investigations will be shovn, among which pictures of multisensor coverage reproducing synthetized graphical images displayed to system users; these images present in real-time the system detection capabilities taking into account the sensor environment and surveillance missions.

In conclusion, this paper shovs that investigations and lessons learned enables us to stretch

out the desing of a future mobile/removeable multisensor system design as well as to estimate its performance and capabilities.

KEY-WORDS

Multisensor tracking - MST - Data Fusion - Multiple Plot - Variable Update - MP-VU - Multisensor Management.

SUMMARY

Air situation establishment is a fundamental element in managing the theatre of operation in modern conflicts. Add to this point, the increase in the range of efficiency of weapons as well as in the variety of the threat enlarge the volume in which having air control is a top priority. This leads us to envisage the possibility of moving forward the command and control centres to advanced posts, or allied countries, or of having them follow the conflict area, far from fixed groundbased facilities and sedentary detection means.

This paper point out that air situation establishment includes sensor data collection, plot fusion, smoothing and synthesis of measures delivered by the sensor network in order to manage, control and analyse the manoeuver possibilities of air vehicles. Sensor and fusion centres mobility combined with dramatic shape changes in sensor network both make the fusion process issues more complicated :

- It must have the capability to accept changes in detection coverage and manage them.
- It must have the capability to compute and correct quickly sensor alignment errors.
- It must have the capability to process data obtained in jamming conditions.

Operational realisations done by THOMSON-CSF enable us to stretch out the design of a future mobile/removable multisensor system as well as to estimate its performance and capabilities.

1. INTRODUCTION

Keeping control of the air situation is a fundamental element in managing the theatre of operations in modern conflicts. The recent war in the Golf illustres very well the high priority of air operation control.

The master is essential no only for offensive missions but also to provide own territory defense.

Most of the command and control systems, well known under the C^2 acronym, have been designed to carry on operations on national territories, or at least in country's areas of interest or areas of responsibility. Changes in threat for East-West to North-South conflicts as well as "internationalization" of crisis and extension of the defense requirements out of the national boundaries to save economical alliances both lead to think about mobile command and control centres (or at least part of C^2 centres). These centres would move :

- on advanced posts,
- on allied countries,
- close to the forward edge of the battle areas,
- far from fixed facilities and sedentary detection and communication means.

Several items contribute to enlarge the volume in which having air control is a top priority. These are :

- the increase in range of efficiency of modern weapons (i.e. balistic missiles, cruise missiles in flight, fighter re-fueling, ...),
- the use of a large variety of sensor,
- the cooperation and data exchange between command and control systems.

2. PURPOSE

In this nev environment, mobile command and control systems/centres will have to gather sensor data from removeable and mobile detection means in order to produce an accurate and understandable air situation. Then, the recognized air picture will be disseminated to the users to support air vehicle guidance and air missions. So, the surveillance is to be considered as a "service" providing data to the other command and control functions.

Main tasks of the surveillance function are :

- produce and maintain the recognized air picture,
- manage dedicated surveillance means,
- surveillance data exchange and dissemination.

This points out that multisensor data fusion embraces sensor data collection, plot fusion, smoothing, identification and synthesis of measures delivered by the sensor network and other detection subsystems in order to assess the air picture. Main components of the surveillance are :

- sensors and dedicated detection sub-systems,
- communication means,
- data fusion centres.

Target information is obtained with or without cooperation of the air vehicle, and result from active or passive measures ; it can include identification data and classification of the target.

Sensors and fusion centres mobility combined with dramatic shape changes in sensor network both make the fusion process issues more complicated. The most important concerns due to mobility are the following :

- Sensor concerns : mobile or transportable sensors generally have a lower capability and performance than groundbased series (lover transmissions power, smaller antenna size, design adapted to mobility, reinforced hardware and electronics).
- Alignment concerns : mobility introduces unaccuracy on sensor location and on verticality and north alignment. Trivial means will be commonly used to set up sensors and to align them on site when the operating area is out of the coverage of more accurate localisation system such as satellites (global positionning system for instance).
- Communication concerns : data exchange capa-

bility and data quality and reliability are mainly in correlation with the data links. Due to mobility constraints (but also jamming conditions), rate and characteristics of sensors data links may be of low fidelity.

- Concern on the air picture consistency : the data fusion systems will have to handle smooth changes in sensors number and positions.

3. THOMSON-CSF EXPERIENCE

THOMSON-CSF has completed a multisensor tracker based on the Multiple Plot Variable Update technique (MST/MP-VU). This technique embraces numerous advantages due to direct plot/strobe processing. Industrial choices and company design for on-site implementation and system integration allow this product to solve a large amount of concerns here above described in this paper.

The problems to solve are :

- capability to accept changes in detection coverage and to manage gaps in the coverage,
- ability to compute and correct on-line sensor parameters (i.e. biases and misalignment),
- design to fuse data from a broad range of sensor types which have different characteristics (measures accuracy, working modes, number and type of reported data, ...),
- display of the current multisensor coverage as a sensor performance,
- capability to process data obtained in jamming condition.

The following sections describe the nature of these concerns and the way to solve them.

3.1. Systematic error assessment

A prerequisite for efficient multisensor tracker is to handle the registration problem. Any attempts to implement a multisensor tracker with poor sensor bias estimation will result in an augmentation of correlation gate sizes which will jeopardize benefits in their reduction achieved by small time intervals between updates. Furthermore, the adding of a considerable amount of information under such conditions might result in a number of improper correlations.

Errors are due to the systematic errors that differ for each sensor (macro type errors). The macro type errors are radar location error, elevation bias, azimuth bias, range bias, time-stamping errors, squint errors and verticality errors. Such biases are especially critical in multisensor tracking, since they cause different track trajectories swings that are accentuated by a small spacing of track updates and appear as if the track was pulling very high acceleration between updates.

Macro errors may change abruptly in time due to technical maintenance, due to the influence a changing wind direction has upon mechanics of a radar antenna and due to wrong relocation of a mobile radar. This requires dynamic estimation of these macro errors, on-line with the maintenance of tracks.

Then the systematic radar error assessment estimates on-line in a multiradar environment with sufficient radar overlap the systematic errors per radar :

- radar location errors, due to wrong location of the radar,
- elevation bias, due to wrong elevation adjustment of the antenna,
- azimuth bias, due to wrong north adjustment of the antenna,
- range bias, due to wrong zero kilometer adjustment of the radar,
- time-stamping bias, due to systematic errors of the plot time-stamping (time in storage, transmission time),
- squint, due to plot extraction for radars with antenna diagram tilted with squint angle,
- radar verticality, due to unverticality of radar rotation axe.

Position and attitude bias of different sensors are treated as state vectors of an estimator and are updated each time the sensor reports a new measurement which correlates with one of the selected tracks. To optimize the choice of the data used by the filter, a selection of tracks useful for the filter is made. This selection is based on criteria like the number of radars which detects the track and the kinematic of the track.

3.2. Operation in Jamming Condition

Fielded systems based on the MP-VU technique include a strobe processing capability. Plots from detection sources may have two or three dimensions according to the radar sensing capacity, but there are also one - or two dimensional data (strobes) delivered by the radar (ECM mode), by an ESM sensor, or by a passive jammer locater sensor, when a target starts to jam or emit signals.

THOMSON-CSF, based on its experiences, advocates a multisensor architecture with fusion of active and passive data. It is based on :

- a direct correlation of strobes with existing tracks,
- a track initiation with a deghosting function on residual strobes (with an operator help for this process),
- a track updating process using either the plots or strobes.

Actually, this is the best possible approach satisfying the following requirements :

- passive data process able to cope with a reduced sensor deployment,
- resistant against blink jamming,
- exhaust information from every report,
- minimize operator workload for fusion process,
- smooth transition from active to passive tracking.

All this is possible because we have the capacity to update a track by using a filter which is provided with a measurement sharing loop, which means that, unlike a conventional filter (or bank of filters), partial azimuth, range or elevation measurements can be introduced in the filter.

So, THOMSON-CSF has made possible automatic

updating of multisensor tracks with :

- "normal" plots only,

- "normal" plots and strobe plots,

- exclusively strobe plots.

3.3. Sensor Management

The increase of threat variety and performance require system adoption to theater constraints and situation and need to exploit the full spectrum of sensors diversity performance and adaptativity features. This cannot be done manually so there is a need for an automatic multisensor management function. This function operates on operational directives which are expressed in such a way :

- improve surveillance in such and such areas,

- operate in discrete mode (still maintaining sufficient quality of the air picture),

- increase detection capability for small RCS targets in such penetration corridor,

- implement emission plan, which may lead to reduce the use of few sensors in defined sectors.

However, very direct orders can still be issued at this level, such as :

- stop radar emission or low power emission (in front of ARM attack for instance),

- set a high priority quality in the updating for specifically designated tracks.

To fulfill its role this function needs to evaluate the multisensor coverage. This evaluaton function operates from information provided by different sources : from the geographical configuration of the system (ground, sensors locations), from the environmental data provided by the radars or by the operator and evaluates the quality of the multisensor coverage of the system (in real time or in an off-line prediction mode).

3.4. MST/MP-VU Advantages

Mainly the THOMSON-CSF's MP-VU tracker features the following :

- broad use of the sensor coverage redundancy through the full processing of every sensor data as soon as it is reported,

- very high level of performance reached : short automatic track initiation, quick target manoeuver detection and accurate track position during air vehicle changes in speed, heading and altitude,

- benefit from the increase of target detection probability provided by the multisensor coverage,

- ability to process a broad variety of sensor data such as 3D and 2D plots with or without range rate, strobe data, with associated measurement errors,

- robustness against jamming thanks to partial measurement processing capability (track update on strobe data, process of intersection of strobes from non-colocated sensors),

- automatic sensor monitoring, assessment of sensor biases and adjustment of sensor data concurrently with the air picture production.

The high level of performance reached has been validated by tracking results obtained from the data on real sites where THOMSON-CSF has integrated the MST/MP-VU technique. From the operational experience on one hand and from studies in process on existing airborne systems of future ones on the other hand, THOMSON-CSF prepares new solutions based on a distributed network of fixed and mobile sensors.

4. POSSIBLE EVOLUTIONS

Air vehicles are both part of the air picture and users of this picture. When secure high quality communication means will be commonly integrated on board, then air vehicles participate to the air picture production sending their own detection measures, because this data will be very accurate time stamped and well located (MIDS data for instance). Furthermore, these air vehicules would receive at filtered air picture back from the ground based multisensor tracking. So, their local situation will be increased and their own mission will be modified by the ability to start transmitting on board sensors at the last phase.

5. CONCLUSION

In conclusion, when such an implementation of the surveillance fonction and its relevant advantage is compared to the concerns relative to mobility of air surveillance systems, the following matrix appears. It underlines the aproprietaness of this solution for the air situation establishment in a multisensor environment.

Less poverfull sensors
∧
|
∨

- Benefit of the multisensor coverage integration of various sensors and airborne sensor systems

Unaccurate sensor location
∧
|
∨

- Automatic sensor registration/alignment
- Use of air vehicle position reports compared to track/target position

Changes in sensor network (in number and in shape)
∧
|
∨

- MP-VU is monosensor for each set of data/track update
- Sensor management

REFERENCES

- M.Carpentier (1984), "Radars, Bases Modernes", Masson.
- E.Roubine, J-Ch.Bolomey, S.Drabovitch and C.Ancona (1978), "Antennes", Masson.
- F. Le Chevalier (1989), "Principes de Traitement des Signaux Radar", Masson.

- A.Farina and F.A.Studer (1986),
  "Radar Data Processing",
  Research Studies Press LTD.
- Dr J.Llinas and D.Hall (1989),
  "Data Fusion",
  State of the Art LTD.
- S.S.Blackman (1986),
  "Multiple-Target Tracking with Radar
  Application",
  Artech House INC.
- M.Desbois (1987),
  "Multiple Radar Tracking and real time
  processing",
  Milcomp'87 in London U.K., sep 87.
- M.Desbois (1988),
  "Multisensor Tracking",
  Signal Magazine, Nov 88.
- M.Desbois (1990),
  "Detection means management in a multisensor
  environment",
  Radarcon'90 in Adelaide Australia, Apr 90.

SYMBOLS AND ACRONYMS

ARM	Anti-Radiation Missile
C^2	Command & Control
ECM	Electronic Counter-Measure
ESM	Electronic Support Measurement
MIDS	Multifunction Information Distribution System
MP-VU	Multiple Plot-Variable Update
MST	MultiSensor Tracking
PJL	Passive Jammer Locator.

Situation Assessment in the Paladin Tactical Decision Generation System

John W. McManus
NASA Langley Research Center
Hampton, Virginia

Alan R. Chappell
Lockheed Engineering and Sciences Company
Hampton, Virginia

P. Douglas Arbuckle
NASA Langley Research Center
Hampton, Virginia

SUMMARY

Paladin is a real–time tactical decision generator for air combat engagements. Paladin uses specialized knowledge–based systems and other Artificial Intelligence (AI) programming techniques to address the modern air combat environment and agile aircraft in a clear and concise manner. Paladin is designed to provide insight into both the tactical benefits and the costs of enhanced agility. The system was developed using the Lisp programming language on a specialized AI workstation. Paladin utilizes a set of air combat rules, an active throttle controller, and a situation assessment module that have been implemented as a set of highly specialized knowledge–based systems. The situation assessment module was developed to determine the tactical mode of operation (aggressive, defensive, neutral, evasive, or disengagement) used by Paladin at each decision point in the air combat engagement. Paladin uses the situation assessment module and the situationally dependent modes of operation to more accurately represent the complex decision-making process of human pilots. This allows Paladin to adapt its tactics to the current situation and improves system performance. This paper discusses the details of Paladin's situation assessment and modes of operation. The results of simulation testing showing the error introduced into the situation assessment module due to estimation errors in positional and geometric data for the opponent aircraft are presented. Implementation issues for real–time performance are discussed and several solutions are presented, including Paladin's use of an inference engine designed for real–time execution.

1 INTRODUCTION

Modern air combat simulations require an intelligent system, called a Tactical Decision Generator (TDG) , to select the combat maneuvers to perform throughout an air combat engagement. The simulation system must also have the ability to model agile aircraft. The system should have a modular software structure so that new weapons systems or aircraft subsystems (e.g. sensors or propulsion systems), modifications to aircraft control systems, or changes to the aircraft configuration can be easily incorporated. In support of the study of superagile aircraft at Langley Research Center (LaRC), a Tactical Guidance Research and Evaluation System (TiGRES) is being developed. The design and development of TiGRES as well as its relationship to other current air combat simulation systems is described in detail in references [1,2,3].

The TiGRES system is designed to allow researchers to develop and evaluate aircraft systems in a tactical environment. The three main components of TiGRES are a TDG, the Tactical Maneuver Simulator (TMS), and the Differential Maneuvering Simulator (DMS). Both the TMS and the DMS use a TDG as the intelligent automated opponent.

The TMS[1,3] provides a high-fidelity batch air combat simulation environment for the development and testing of various guidance

and control strategies. The researcher defines the initial conditions of the air combat engagement and the TMS then controls the trajectories and attitudes of the aircraft using simple trajectory commands, or through a tactical decision generation system. The main elements of the TMS are a high-fidelity, nonlinear six degree-of-freedom (d.o.f.) rigid-body aircraft dynamic model, including the control system, a TDG, and a user interface.

The DMS consists of two 40' diameter domes and one 20' diameter dome. The facility is intended for the real-time simulation of air combat engagements between piloted aircraft. By using a TDG to control one of the airplanes, it is possible to test a TDG against a human opponent. This feature allows the guidance logic to be evaluated against one or more unpredictable and adaptive human opponents.

2 THE PALADIN SOFTWARE

Paladin is a knowledge-based TDG designed to provide insight into both the tactical benefits and the costs of superagility. Knowledge-based systems use a large amount of information about a problem's domain to understand and solve that problem. Paladin was developed in the Lisp programming language using a Symbolics 3650† workstation.

The development of Paladin has been a multi-stage process using a baseline version of the Adaptive Maneuvering Logic [4] (AML) program as the starting point. Paladin uses the trial maneuver generation and evaluation concept outlined in the AML program [4] with several extensions. The original set of five to nine aircraft trial maneuvers used by AML has been replaced with four sets of positionally dependent trial maneuvers. Past research [2,3] has shown that the use of the positionally dependent sets of trial maneuvers improves overall system performance, allows Paladin to perform target acquisition and tracking more effectively, and improves Paladin's defensive and evasive maneuvering performance.

† Symbolics 3650 is a registered trademark of Symbolics Incorporated

Paladin uses an object-oriented programming approach [5] to represent each aircraft in the simulation. Each aircraft object includes information on the current state of the aircraft's offensive systems (e. g. guns, missile systems, fire control radars, etc.), defensive systems (e. g. electronic counter–measures, chaff, etc.), and propulsion system. This state information is used to help guide Paladin's reasoning process.

Paladin utilizes modular software subroutines and specialized computer hardware. The separation of the aircraft simulation and decision logic components, and the use of highly specialized knowledge sources, allows each module or knowledge source to be designed and implemented using the hardware and programming techniques specifically suited for its function. The use of highly specialized and independent knowledge sources also provides for modular protection[5], confining the effect of an error occurring in a module at run-time to that module, or to a small set of neighboring modules in the program. The confining effect of the modular protection was used to aid in the design and debugging process. Each knowledge source was developed and tested independently before it was incorporated into Paladin.

The independence of the knowledge sources also increases the efficiency of Paladin by allowing knowledge sources to be distributed across a network of several heterogeneous processors. The network currently consists of a Symbolics 3650 workstation, a Symbolics MacIvory† workstation, two SUN† SPARC class workstations, and four Vax 3200† class workstations. Communication between the distributed knowledge sources is achieved using customized DECNet–based Client/Server software developed in–house for TiGRES. This software allows for

† MacIvory is a registered trademark of Symbolics Incorporated.

† SUN is a registered trademark of Sun Microsystems Incorporated.

† Vax 3200 & DECNet are registered trademarks of Digital Equipment Corporation.

synchronization, communications, and data sharing between heterogeneous computers running the DECNet communications protocol. TCP/IP based communications software has also been developed in–house for the SUN workstations. Paladin is implemented as a serial blackboard system, so no serialization or concurrency related software is required[6]. Each knowledge source requests all of the data required to perform its computation from the blackboard at the start of its execution cycle, and posts its results to the blackboard at the end of its execution cycle.

2.1 The Paladin Inference Engine

The Paladin knowledge sources use a custom inference engine (see appendix A) that was designed to support real-time execution of knowledge–based systems. The inference engine uses a depth–first evaluation strategy [5] to search the active rule–bases. Rule–bases can be partitioned, and the partitions are linked using meta–rules (rules used to guide the activation of rule–bases). Rule–bases are expressed in two formats: interpreted lists of condition action pairs used during the design stage, and compiled lists of in-line function definitions used in the final, real–time version of the system. The interpreted lists are used to develop and debug the initial versions of the rule–base. Most existing rule-based systems stop development at this point and implement the interpreted rule–bases. The use of the interpreted rules severely limits the execution performance of the inference engine, and restricts the real-time usage of this type of system. To overcome this problem and allow real–time execution, the rule–base is "compiled" into a list of in–line functions. The rule–base compiler was developed in–house and is written in Lisp for execution on an AI workstation. The compiled rule–bases execute approximately 90 to 100 times faster than the interpreted rules. The inference engine executes a representative test rule–base consisting of 40 rules in the interpreted format in 170 milliseconds. The inference engine executes the same rule–base in the compiled format in 1.9 milliseconds.

There is a trade–off between the length of the rule–base's longest execution path and the knowledge source's execution time. The shorter the execution path is, the faster the execution time. The rule–bases used by Paladin have been partitioned to increase system performance by grouping related rules into small partitions and using meta–rules to link the partitions. This partitioning decreases the number of rules that are active, and decreases the length of the worst–case execution path through the rule–base. The rule–base partitioning allows the designer to calculate the longest and shortest path through the rule–base and compute both a maximum and minimum knowledge source execution time. The knowledge source's maximum execution time can be used to insure that the system will meet real–time execution requirements. If the maximum execution time exceeds the allocated execution time, the designer may be able to repartition the rule–base until real–time execution requirements are achieved.

Paladin uses two rule–bases: a mode selection rule–base used by the Situation Assessment knowledge source, and a throttle control rule–base used by the Active Throttle Controller. The mode selection and throttle control rule–bases are included in appendices B and C. The mode selection rule–base consists of four partitions and contains nineteen rules. The shortest execution path in this rule–base results in a single rule being fired; the longest path results in twelve rules being fired. The throttle control rule–base consists of ten partitions and contains forty rules. The shortest execution path in this rule–base results in a two rules being fired; the longest path results in thirteen rules being fired.

2.2 Situation Assessment Module

Six modes of operation, shown in table 1, have been incorporated in Paladin. As shown in Figure 1, the Situation Assessment knowledge source is executed before the maneuver scoring module. The Situation Assessment knowledge source uses the mode selection rule–base (appendix B) to determine the system's current mode of operation. This knowledge source is used to model a pilot's situational awareness and changing problem–solving strategies. Just as a pilot will

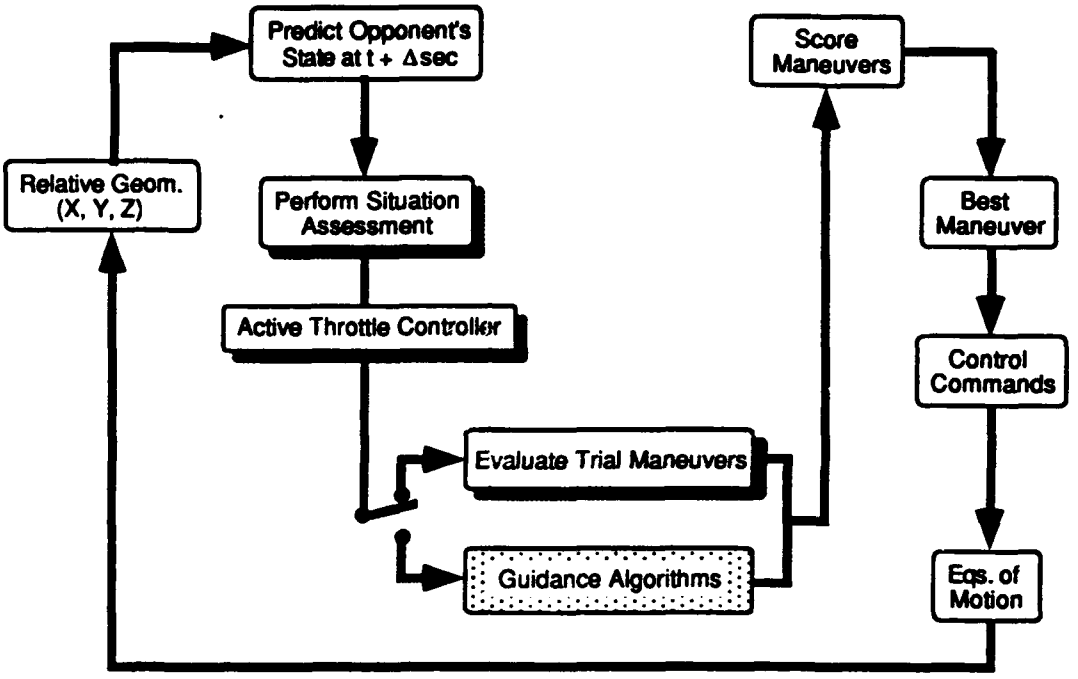


Figure 1. Schematic Of Paladin

recognize the difference between an aggressive situation and a evasive situation and react accordingly, the Situation Assessment knowledge source provides information allowing Paladin to adapt its problem–solving strategy based on the current situation. The determination of the current mode of operation is based on the aircraft's current mission, the current state of the aircraft's systems, the relative geometry between the aircraft and its opponent, and the opponent's instantaneous-*intent* (defined later). Each of the six modes of operations has a unique vector of scoring weights and a unique decision interval (shown in table 1). The scoring weights[2] for each mode of operation have been adjusted during the design and testing process to maximize Paladin's performance in that mode of operation. The testing procedures used to evaluate Paladin's performance are described in detail in section 3.

Both TMS and DMS test results[1,2] have shown that a short decision interval (0.25 sec.) improves Paladin's fine–tracking performance in aggressive situations, and Paladin's maneuvering capabilities in evasive situations. In neutral or defensive situations the same short decision interval results in a "thrashing" motion, degrading system performance. The thrashing is due to the system overcompensating for small changes in the opponent's motion. These thrashing maneuvers bleed off energy and reduce Paladin's effectiveness; thus a longer decision interval is used in defensive (0.5 sec.) and evasive situations (1.0 sec.).

Table 1. Modes of Operation

Mode	Decision Interval
Aggressive	0.25 sec
Defensive	0.5 sec
Evasive	0.25 sec
Ground Avoidance	0.125 sec
Neutral	1.0 sec
Disengage	0.5 sec

The situation assessment knowledge source also determines the opponents instantaneous-intent. The opponent's instantaneous-intent is defined to be an estimation of the opponent's mode of operation at the current point in time based on Paladin's available sensor, positional, and

geometric data. Currently, there is no attempt to use a history of instantaneous-intent to derive a long-term opponent intent.

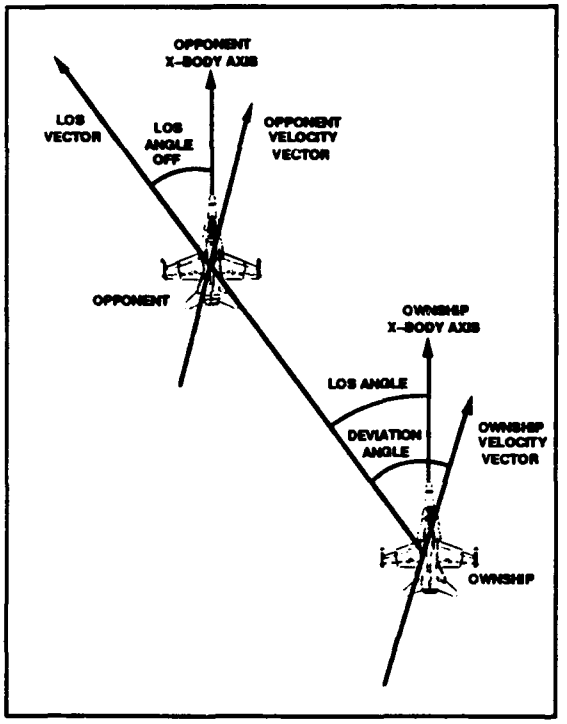


Figure 2. Angle Definitions

2.2.1 *Situation Assessment Data*

To perform situation assessment, information on aircraft relative geometry and Paladin's system status is required. This information is available in the form of participant-specific data maintained by Paladin. All data relating to the Paladin aircraft as well as Paladin sensor data (e.g. the opponent's relative position) are assumed to be known exactly. Other data required about the opponent must be estimated.

The quantities used by the situation assessment module which are based on exactly known data are either specific to the Paladin aircraft or are relative values from the Paladin aircraft's point of view. Paladin's current throttle position and altitude are parameters taken directly from the current state. Range is the magnitude of a vector connecting the centers of gravity of the aircraft. The Line-Of-Sight (LOS) angle is defined as the angle between the LOS vector and the ownship body x-axis (see figure 2); the deviation angle is defined as the angle between the LOS vector and the ownship velocity vector; and the LOS angle off is defined as the angle between the LOS vector and the opponent's body x-axis.

The deviation angle is calculated as the inverse cosine of the magnitude of the projection of the range onto the velocity vector divided by the range. In equation form,

$$\text{deviation angle} = \arccos\left[\frac{\dot{x}\Delta x + \dot{y}\Delta y + \dot{z}\Delta z}{(\text{Range})\,|\text{Velocity}|}\right], \quad (1)$$

The line-of-sight angle (LOS) is the inverse cosine of the magnitude of the projection of the range onto the x-body axis divided by the range, or,

$$\text{LOS angle} = \arccos\left[\frac{D(1,1)\Delta x + D(1,2)\Delta y + D(1,3)\Delta z}{\text{Range}}\right], \quad (2)$$

where Δx, Δy, and Δz represent the difference between the two aircraft positions. D(i,j) is the i, j element of the Paladin body axis direction cosine matrix. Then the LOS elevation is taken to be the inverse sine of minus the opponent's z-coordinate in the Paladin body axis system divided by the range, or,

$$\text{LOS elevation} = \arcsin\left[\frac{-z_{\text{opponent in Paladin body axis system}}}{\text{Range}}\right]. \quad (3)$$

The LOS azimuth is the inverse tangent of the opponent's y-coordinate divided by the opponent's x-coordinate, both in the Paladin body axis system,

LOS azimuth =

$$\arctan\left[\frac{y_{\text{opponent in Paladin body axis system}}}{x_{\text{opponent in Paladin body axis system}}}\right]. \quad (4)$$

Finally, off corner is the proportion by which Mach number differs from the instantaneous cornering Mach number (speed to achieve largest possible turn rate) calculated for the current altitude,

$$\text{off corner} = \frac{\text{Cornering Mach - Current Mach}}{\text{Cornering Mach}}. \quad (5)$$

The velocity, acceleration and orientation of the opponent must be estimated, since this data would not be available from sensors. These estimates are made using a three point time history of the known position data and several assumptions about the opponent aircraft (weight, wing surface area, and flight characteristics). The current position of the opponent and the opponent's position at the preceding two decision cycles are used to find a quadratic curve fit for the position as a function of time. The first and second derivatives of this function at the current time yield an estimation of the opponent's instantaneous velocity and acceleration. By assuming aerodynamic characteristics of the opposing aircraft, and using the velocity and acceleration estimates, an estimated body-axis orientation for the opponent can be found.

The quantities used by the situation assessment module which are based on estimated data are largely relative values from the opponent aircraft's point of view. Each of these quantities has some error introduced by the estimation process. The range rate is the magnitude of the projection of the relative velocity onto the range axis (all in the inertial axis system),

$$\text{range rate} = \frac{\dot{\Delta x}\Delta x + \dot{\Delta y}\Delta y + \dot{\Delta z}\Delta z}{\text{Range}}. \quad (6)$$

The opponent's deviation angle and line-of-sight angle are calculated similarly to the Paladin aircraft parameters (equations 1 and 2), using the opponent's velocity and x-body axis. Paladin's LOS angle off is defined as 180° - opponent's LOS angle. The errors between the actual parameter values and the estimated values were calculated for a representative set of 32 within–visual-range engagements[3], resulting in the sample means ($\bar{X}$) and standard deviations (s) listed in table 2 (for a sample size of 7369). Figures 3 through 5 show each of these error magnitudes (absolute value of the actual value minus the estimated value) during the course of a typical engagement. Error expectations given in table 2 and magnitudes given in the figures are based on engagements between Paladin and an opponent with known aerodynamic characteristics. If the aerodynamics of the opponent aircraft are not well known, the error in the LOS angle should increase, since this error is strongly dependent on the aircraft flight characteristics.

From the same set of 32 engagements, results were collected on the sensitivity of the situation assessment module to these estimation errors. The correct mode of operation (the mode selected given exact data for all inputs) was chosen 98.0% of the time. Hence, this implementation of the situation assessment knowledge source is believed to be insensitive to the errors introduced by data estimation. Past research[2] has shown Paladin to be insensitive to errors due to typical levels of sensor noise in the opponent's positional data.

Table 2. Error Statistics

Range Rate Error (ft/sec)		Deviation Error (deg)		LOS Error (deg)	
$\bar{X}$	s	$\bar{X}$	s	$\bar{X}$	s
1.52	0.68	2.06	0.13	8.02	3.05

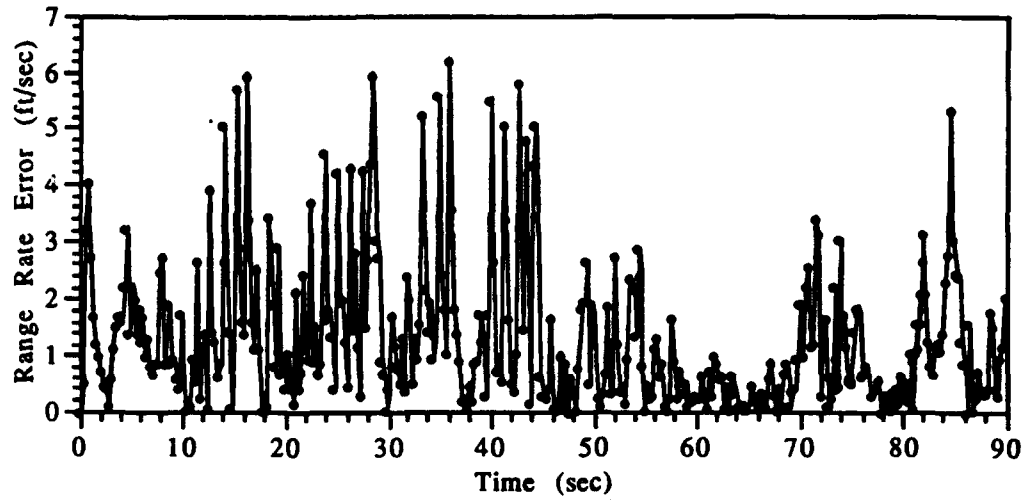


Figure 3 Range Rate Error during Engagement.

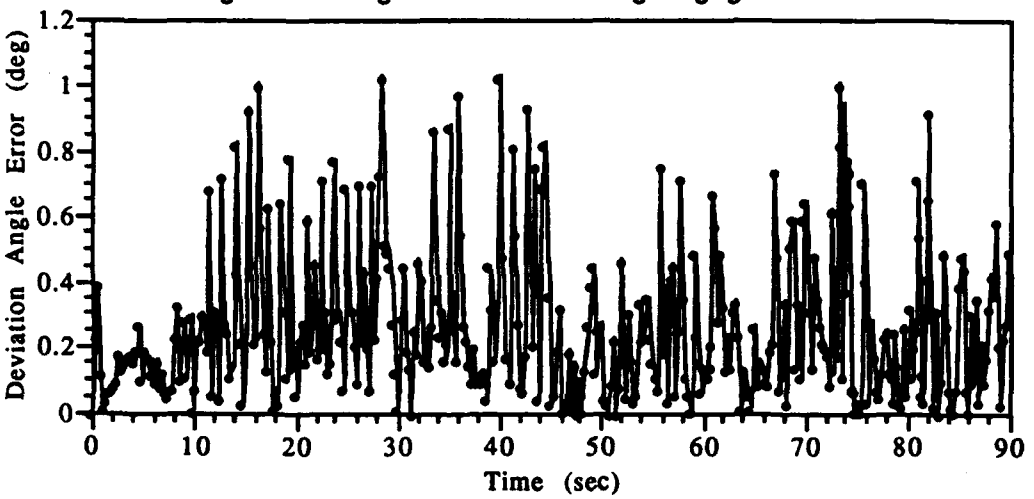


Figure 4 Deviation Angle Error during Engagement.

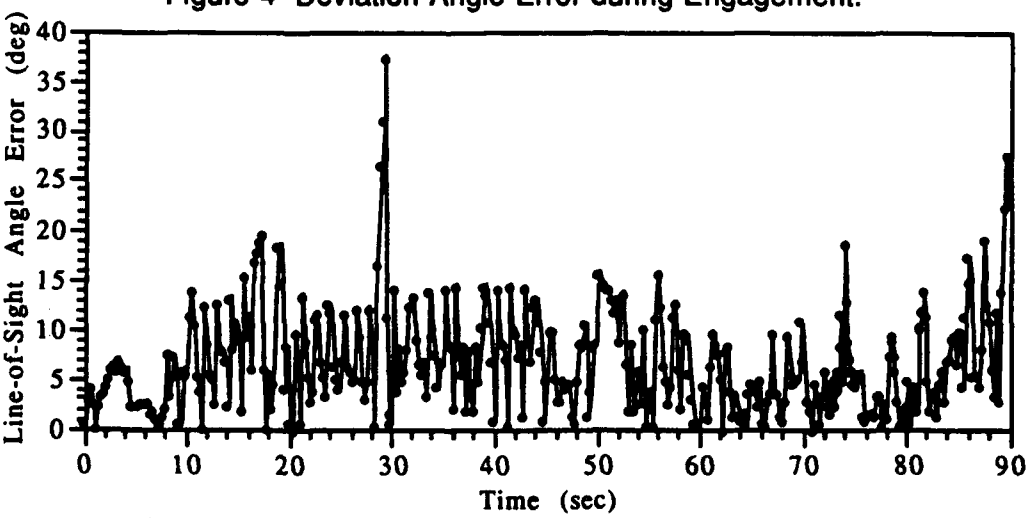


Figure 5 Line-of-Sight Angle Error during Engagement.

2.3 Active Throttle Controller

A rule-based Active Throttle Controller was developed to adjust the throttle setting based on the current mode of operation. The throttle controller is called at the start of each decision interval and can set the throttle to any position between idle and full afterburner [0.0 = flight idle, ..1.0 = military power, ..2.0 = full afterburner]. The throttle controller uses the throttle control rule–base (appendix C), the current mode of operation, and the relative geometry information to select either a target acquisition mode, a fine tracking mode, or a target or missile avoidance mode. Each mode has a set of specific throttle control rules that are used to maximize system performance in that mode.

The active throttle controller uses the same data described for the situation assessment module, and so, incurs the same estimation errors. Using the results of the 32 engagements discussed in the previous section, the resulting errors in the selected throttle position have been evaluated. The throttle setting chosen by the active throttle controller was within +/-5% of the correct setting (the position selected given exact data) in 95.8% of the 7369 cases tested. Table 3 shows the distribution of these errors around the correct throttle command. For this table, E is the error band (in %) of the throttle position, and P is the percentage of the test cases which fall within +/- E of the correct position.

Table 3. Active Throttle Controller Error

E	P
5	95.81
10	95.83
15	95.90
20	97.15
50	99.38

2.4 Maneuver Scoring Module

The Paladin Maneuver Scoring Module knowledge source is a FORTRAN subroutine. The scoring module uses a set of fuzzy logic questions[3] with responses ranging from [-1.0 = Negative, ..0.0 = Neutral, ..1.0 = Positive] and the mode-specific scoring weight vector selected by the situation assessment module to score each of the trial maneuvers. For each trial maneuver evaluated the predicted positions for both the opponent and the Paladin aircraft are computed. The position of the opponent is extrapolated using a quadratic curve fit based on the time history of the opponent aircraft's trajectory as previously described. The future position of the Paladin aircraft is determined by predicting the result of executing the control commands for each candidate trial maneuver.

Once the relative geometry between the future positions of the two aircraft is calculated, the score for the maneuver is determined by computing the responses to the seventeen fuzzy logic questions, applying the selected scoring weight vector, and then summing the results to generate a single numeric score. After all of the trial maneuvers have been evaluated, the highest scoring maneuver is selected and the associated control commands are executed.

3 Paladin Testing Procedures

Paladin is currently being tested in the TMS using six d.o.f. aircraft dynamics, and in the DMS using five d.o.f. aircraft dynamics[2,3]. TMS testing is done in a non-real-time, batch mode environment against a baseline TDG. Each group of test conditions consists of 32 sets of initial aircraft conditions. The initial altitudes, airspeeds, and the separations between the two aircraft are adjusted to provide representative coverage of the within–visual–range air combat arena. The largest initial aircraft separation currently being tested (5 nm) places the aircraft at the transition point between beyond-visual-range and within-visual-range air combat.

A set of engagement scoring metrics (presented in reference 3, and outlined in Appendix D) are reviewed after each set of test runs and the data are used to tune the mode specific scoring weights and test the completeness of the knowledge bases. Although the metrics are helpful, no single metric has been developed that can completely measure the performance of an aircraft in the

engagement. In past test engagements an aircraft could score significant amounts of weapons lock time after it had been "killed." This phenomenon adversely affected several of the scoring metrics. To correct this problem all engagements are now ended when the probability of survival for either aircraft is less than 0.30.

After initial adjustment of the scoring weights, the set of initial conditions is expanded to 320 initial conditions by modifying the initial separation between the airplanes, the initial altitudes, and the initial Mach numbers. This stepwise refinement process provides the large set of results required to achieve global system improvements across the total within–visual–range air combat environment.

A baseline version of Paladin is currently being tested in the DMS using a 5 d.o.f. aircraft model. The aircraft model lacks both the extra degree of freedom (lateral motion in body axes) as well as an accurate representation of the aircraft's rotational dynamics throughout the complete flight envelope. The baseline Paladin system, the Computerized Logic for Air Warfare Simulation (CLAWS) contains the situation assessment module, the active throttle controller, and a reduced set of situationally dependent trial maneuvers. This reduced set of trial maneuvers and the simplified aircraft model are used to insure real-time performance in the DMS.

The development of CLAWS has made it possible to evaluate the tactical decision generation software against human pilots in the DMS in a realistic air combat environment. This capability has allowed experienced pilots to interact with the system and comment on its performance and suggest improvements. The pilots' comments and suggestions are then the basis for changing the TMS experimental version of Paladin. These changes are tested and refined before being included in the baseline system.

4 Concluding Remarks

Paladin, a computerized air combat tactical decision generator, has been developed to study within-visual-range air combat engagements. The system incorporates modern airplane simulation techniques, sensors, and weapons systems. The system was developed using several concepts first outlined in the Adaptive Maneuver Logic program. Paladin uses knowledge–based systems and Artificial Intelligence (AI) programming techniques to address air–to–air combat and agile aircraft in a clear and concise manner. The ability to integrate Paladin into the Differential Maneuvering Simulator offers a unique opportunity to evaluate the performance of the AI-based Paladin software in a real-time tactical environment against human pilots.

Paladin models aspects of the decision–making processes used by human pilots through the use of the Situation Assessment knowledge–based system. The use of distinct modes of operation allows Paladin to perform complex air–to–air combat tasks and generate sound tactical decisions in real–time. Paladin presents an excellent opportunity to evaluate the use of AI programming techniques and knowledge-based systems in a real-time environment.

References

1. Goodrich, Kenneth H; McManus John W. : *"Development of A Tactical Guidance Research and Evaluation System (TiGRES)."* AIAA Paper #89-3312, August 1989.

2. McManus John W.; Goodrich, Kenneth H. : *"Application of Artificial Intelligence (AI) Programming Techniques to Tactical Guidance for Fighter Aircraft."* AIAA Paper #89-3525, August 1989.

3. Goodrich, Kenneth H; McManus John W. : *"An Integrated Environment For Tactical Guidance Research and Evaluation."* AIAA Paper #90-1287 May 1990.

4. Burgin, G. H., et al.: *An Adaptive Maneuvering Logic Computer Program for the Simulation of One-on-One Air-to-Air Combat.* Vol I and Vol II. NASA CR-2582, CR-2583, 1975

5. Meyer, Bertrand. *Object-oriented Software Construction*. Ed. C.A.R. Hoare. Prentice Hall International Ltd, 1988.

6. McManus John W.: "A Parallel Distributed System for Aircraft Tactical Decision Generation" In *Proceedings of the 9th Digital Avionics Systems Conference*, 1990, pp. 505 – 512

Appendix A. The Paladin Inference Engine

The inference engine used by Paladin is designed for real-time execution of complex rules. The inference subroutine accepts a rule–list as input. A rule–list consists of a set of compiled in-line functions developed from the rule–base. Each rule consists of a condition -> action pair. The condition clause of a rule can be any function that returns a boolean value of TRUE or FALSE. The action clause of a rule can be any computable function, including calling the inference engine with another rule–list. The Inference subroutine evaluates the condition clause of the first rule in the rule-list. If the condition evaluates to TRUE the action clause is executed. If the condition is FALSE inference is called recursively with the remaining elements in the rule–list.

(subroutine inference (rule-list)
(If rule-list empty -> quit)
(If condition clause of first rule is TRUE -> execute action clause and quit)
(Otherwise -> call inference on the rule-list with the first rule removed))

Appendix B. Paladin Mode Select Rules

The mode select rule–base contains five partitions. The main partition contains ten rules. The offensive–status–true partition contains three rules. The defensive–status–true partition contains two rules. The offensive–status–false partition contains three rules, and the defensive–status–false partition contains one rule. The rules are used to determine the current mode of operation for Paladin and the time interval at which tactical decisions are made.

The rules listed in the mode select rule–base presented here are representative of the rules that are currently used to determine Paladin's mode of operation. This set is not intended to be all inclusive or definitive.

The selection process is started by sending the rule-list to the inference engine shown in Appendix A.

(Call inference mode-select-rules)

(rule-list mode-select-rules (mode-rule-1 mode-rule-2 mode-rule-3 mode-rule-4 mode-rule-5
mode-rule-6 mode-rule-7 mode-rule-8 mode-rule-9 mode-rule-10))

(rule mode-rule-1
(If altitude <= 50.0 ft -> set mode to evasive and decision cycle to 0.25))

(rule mode-rule-2
(If relative position is evasive -> set mode to evasive and decision cycle to 0.25))

```
(rule mode-rule-3
  (If distance home >= 461.0 miles -> set mode to disengage and decision cycle to 0.5) )

(rule mode-rule-4
  (If engine status is not perfect -> set mode to disengage and decision cycle to 0.5) )

(rule mode-rule-5
  (If mission is evasive -> set mode to evasive and decision cycle to 0.25) )

(rule mode-rule-6
  (If (mission is defensive and position is aggressive or neutral)
   or (mission is neutral and position is aggressive) ->
      set mode to aggressive and decision cycle to 0.25) )

(rule mode-rule-7
  (If offensive and defensive systems are up ->
     Call inference offensive-system-true-list and Call inference defensive-system-true-list) )

(rule mode-rule-8
  (If offensive systems are up ->
     Call inference offensive-system-true-list and Call inference defensive-system-false-list) )

(rule mode-rule-9
  (If defensive systems are up ->
     Call inference defensive-system-true-list and Call inference offensive-system-false-list) )

(rule mode-rule-10
  (If offensive and defensive systems are down ->
     set mode to disengage and decision cycle to 1.0) )

*************************************************************************
Offensive Status True Rules.
*************************************************************************

(rule-list offensive-status-true-list (offense-true-rule-1 offense-true-rule-2 offense-true-rule-3))

(rule offense-true-rule-1
  (If mission is aggressive and position is aggressive ->
     set mode to aggressive and decision cycle to 0.25) )

(rule offense-true-rule-2
  (If mission is aggressive and position is neutral ->
     set mode to aggressive and decision cycle to 0.25) )

(rule offense-true-rule-3
  (If position is neutral -> set mode to neutral and decision cycle to 1.0) )

*************************************************************************
Defensive Status True Rules.
*************************************************************************

(rule list defensive-status-true-list (defense-true-rule-1 defense-true-rule-2))

(rule defense-true-rule-1
  (If mission is aggressive and position is defensive ->
     set mode to aggressive and decision cycle to 0.25) )

(rule defense-true-rule-2
  (If position is defensive -> set mode to defensive and decision cycle to 0.5) )
```

```
*************************************************************************
Offensive Status False Rules.
*************************************************************************

(rule-list offensive-status-false-list (offense-false-rule-1 offense-false-rule-2 offense-false-rule-3))

(rule offense-false-rule-1
   (If mission is aggressive and position is aggressive ->
        (If weapons are functional -> set mode to aggressive and decision cycle to 0.25)
        (Otherwise -> set mode to neutral and decision cycle to 1.0) )

(rule offense-false-rule-2
   (If mission is aggressive and position is neutral ->
        (If weapons are functional -> set mode to aggressive and decision cycle to 0.25)
        (Otherwise -> set mode to disengage and decision cycle to 0.5) )

(rule offense-false-rule-3
   (If mission is neutral and position is neutral ->
        (If weapons are functional -> set mode to neutral and decision cycle to 1.0)
        (Otherwise -> set mode to disengage and decision cycle to 0.5) )

*************************************************************************
Defensive Status False Rules.
*************************************************************************

(rule-list defensive-status-false-list (defense-false-rule-1))

(rule defense-false-rule-1
   (If position is defensive -> set mode to evasive and decision cycle to 0.25) )
```

Appendix C. Paladin Throttle Control Rules

The throttle control rules are used to determine the current throttle position for Paladin. The throttle control rule–base contains nine partitions. The main partition contains three rules. The big–range–list partition contains two rules. The med–range–list partition contains seven rules. The small–range–list partition contains five rules. The corner–list partition contains three rules. The set–range–list partition contains four rules. The push–list partition contains three rules. The hold–list partition contains three rules, and the reel–list partition contains five rules.

The rules listed in the throttle control rule–bases presented here are representative of the rules that are currently used to determine Paladin's throttle position. This set is not intended to be all inclusive or definitive.

The selection process is started by sending the rule-list to the inference engine shown in Appendix A. If the rule currently being evaluated has "TRUE" as the condition clause, the action is always taken.

```
(Call inference throttle-rules)

(rule-list throttle-rules (throttle-rule-1 throttle-rule-2 throttle-rule-3))

(rule throttle-rule-1
   (If range > 20000.0 ft -> Call inference big-range-list) )

(rule throttle-rule-2
   (If 1000.0 ft < range <= 20000.0 ft -> Call inference med-range-list) )

(rule throttle-rule-3
   (If range <= 1000.0 ft -> Call inference small-range-list) )
```

```
**************************************************************************
Big-Range-List
**************************************************************************

(rule-list big-range-list (big-range-rule-1 big-range-rule-2))

(rule big-range-rule-1
   (If I see him -> set throttle = 2.0) )

(rule big-range-rule-2
   (TRUE -> Call inference corner-list) )

**************************************************************************
Med-Range-List
**************************************************************************

(rule-list medium-range-list (medium-range-rule-1 medium-range-rule-2 medium-range-rule-3
                              medium-range-rule-4 medium-range-rule-5 medium-range-rule-6
                              medium-range-rule-7) )

(rule medium-range-rule-1
   (If I see him and he can't see me and range <= 7500.0 ft -> Call inference corner-list) )

(rule medium-range-rule-2
   (If I see him and he can't see me and range > 7500.0 ft -> set throttle = 2.0) )

(rule medium-range-rule-3
   (If I see him and he can't see me and I'm infront of him -> Call inference corner-list) )

(rule medium-range-rule-4
   (If I see him and he can't see me and I'm not infront of him -> Call inference setrange-list) )

(rule medium-range-rule-5
   (If I can't see him and he can see me and he is infront of me -> Call inference corner-list) )

(rule medium-range-rule-6
   (If I can't see him and he can see me and he is not infront of me -> set throttle = 2.0) )

(rule medium-range-rule-7
   (If I can't see him and he can't see me -> Call inference corner-list) )

**************************************************************************
Small-Range-List
**************************************************************************

(rule-list small-range-list (small-range-rule-1 small-range-rule-2 small-range-rule-3
                             small-range-rule-4 small-range-rule-5) )

(rule small-range-rule-1
   (If I'm not infront of him and he is not infront of me -> Call inference corner-list) )

(rule small-range-rule-2
   (If I'm not infront of him and he is infront of me -> Call inference setrange-list) )

(rule small-range-rule-3
   (If I'm infront of him and he is not infront of me and
        (range < 500.0 ft or range rate < 0.0 ft/sec) -> set throttle = 0.0) )

(rule small-range-rule-4
   (If I'm infront of him and he is not infront of me and
        range >= 500.0 ft and range rate >= 0.0 ft/sec -> set throttle = 2.0) )
```

```
(rule small-range-rule-5
   (If I'm infront of him and he is infront of me -> Call inference corner-list) )

*****************************************************************************
Corner–List
*****************************************************************************

(rule-list corner-list (corner-rule-1 corner-rule-2 corner-rule-3))

(rule corner-rule-1
   (If off corner < -0.1 -> set throttle = max (0.5, previous throttle * (1.0 - off corner))) )

(rule corner-rule-2
   (If off corner > 0.1 -> set throttle = previous throttle * (1.0 - off corner)) )

(rule corner-rule-3
   (TRUE -> set throttle = previous throttle)

*****************************************************************************
Setrange–List
*****************************************************************************

(rule-list setrange-list (setrange-rule-1 setrange-rule-2 setrange-rule-3 setrange-rule-4))

(rule setrange-rule-1
   (If range > 15000.0 ft -> set throttle = 2.0) )

(rule setrange-rule-2
   (If 12000.0 ft < range <= 15000.0 ft -> Call inference reel-list) )

(rule setrange-rule-3
   (If 7500.0 ft < range <= 12000.0 ft -> Call inference hold-list) )

(rule setrange-rule-4
   (If range <= 7500.0 ft -> Call inference push-list) )

*****************************************************************************
Push–List
*****************************************************************************

(rule-list push-list (push-rule-1 push-rule-2 push-rule-3))

(rule push-rule-1
   (If separating very fast -> set throttle = previous throttle * 0.95) )

(rule push-rule-2
   (If separating fast -> set throttle = previous throttle * 0.9) )

(rule push-rule-3
   (TRUE-> set throttle = previous throttle * 0.6) )

*****************************************************************************
Hold–List
*****************************************************************************

(rule-list hold-list (hold-rule-1 hold-rule-2 hold-rule-3))

(rule hold-rule-1
   (If separating fast or very fast -> set throttle = max (0.5, previous throttle * 1.05)) )
```

```
(rule hold-rule-2
   (If closing fast or very fast -> set throttle = previous throttle * 0.95) )

(rule hold-rule-3
   (TRUE -> set throttle = previous throttle) )

*************************************************************************
Reel-List
*************************************************************************

(rule-list reel-list (reel-rule-1 reel-rule-2 reel-rule-3 reel-rule-4 reel-rule-5))

(rule reel-rule-1
   (If closing very fast -> set throttle = previous throttle * 0.9) )

(rule reel-rule-2
   (If closing fast -> set throttle = previous throttle) )

(rule reel-rule-3
   (If closing slowly -> set throttle = max (0.5, previous throttle * 1.1)) )

(rule reel-rule-4
   (If separating fast or very fast separating-fast -> set throttle = 2.0) )

(rule reel-rule-5
   (TRUE -> set throttle = max (0.5, previous throttle * 1.2)) )
```

Appendix D. Engagement Scoring Metrics

Four scoring metrics are currently used to evaluate each air combat engagement. All metrics are computed at the aircraft simulation update rate of 32 times per second. The first metric consists of the total time that each airplane has its weapons locked on its opponent, the probability that any weapons fired will hit the opponent, the distance between the opponents, the angle-off, and the deviation angle. The results are printed in a table format at the completion of each run.

The second scoring metric computes a Probability of survival (Ps) using the data computed by the first metric. The probability to hit for an all–aspect missile and for the cannon are computed using the range and LOS angle to the opponent. The probability to hit for a tail–aspect missile is computed using the range, the LOS angle to the opponent, and the LOS angle off. Aircraft missiles are treated as limited resources and a probability to hit of 0.65 is required to launch the first missile. The probability to hit threshold increases by 0.05 for each missile launched. An estimated flyout time (the time it will take a missile to reach it's target) for each missile is computed based on the launch parameters, and another missile cannot be fired until the flyout time has passed. The Ps for an aircraft then is

$$Ps = 1.0 - \Sigma\,[\text{probability to hit} * Ps(f)] \qquad (D.1)$$

summing over each weapon fired by the opposing aircraft. Ps(f) represents the Ps of the aircraft firing the weapon at the time the weapon was fired.

The third scoring metric attempts to determine a Lethal Time (LT) advantage for each

engagement. Lethal time advantage attempts to weigh the lethality of each distinct type of weapons lock time.

$$LT = \frac{\text{Paladin Gun - Opponent Gun}}{2} +$$

$$(2 * (\text{Paladin Tail–Aspect - Opponent Tail–Aspect})) +$$

$$(\text{Paladin All–Aspect - Opponent All–Aspect})$$

(D.2)

A positive lethal time value shows Paladin with a lethal time advantage, and a negative lethal time shows the opponent with an advantage.

The fourth metric is Time on Offense (TOF).

$$TOF = (\text{Gun time} + \text{All-aspect time} + \text{Tail-aspect time})$$

(D.3)

ΔTOF is computed as Paladin's TOF minus the opponent's TOF. A positive ΔTOF value shows Paladin with an time on offense advantage, and a negative ΔTOF shows the opponent with a time on offense advantage.

A TEAMWORK MODEL OF PILOT AIDING: PSYCHOLOGICAL PRINCIPLES FOR MISSION MANAGEMENT SYSTEMS DESIGN

R.M. Taylor
S.J. Selcon
RAF Institute of Aviation Medicine
Farnborough, Hants. GU14 6SZ, UK

1. SUMMARY

Advances in automation and control/display technology have enabled design effort to focus on how aircrew system interfaces can be created to help perform the mission and to help solve mission problems. Evidence of this development towards a concept of aiding the pilot, rather than replacing pilot functions, comes from recent systems proposed for both ground and airborne mission control and management. Interface design solutions for aircrew systems problems usually evolve pragmatically, based on considerations of convenience, availability, utility, familiarity and operator acceptance. Currently, there is no substantial theory of the pilot-aiding concept. In the absence of a theoretical basis, pilot-aiding interfaces for mission control and management will lack theoretical consistency and they will be without any formal, systematic procedures for establishing design criteria, goals and objectives. Some of the consequences of this will be sub-optimal utilisation of system function, loss of situational awareness, a low level of pilot trust, and failure to reduce pilot workload. In order to address the requirements for a theory of the pilot-aiding concept, we propose a model based on the principles of teamwork, where the human and "electronic" crew components work co-operatively towards achieving mission objectives. The teamwork model is derived from the social psychology of small group dynamics, from knowledge of human engineering requirements, and from the need to integrate human resources considerations in system design. Characteristics of the model are incorporated into a prototype tool for auditing the quality of interface design solutions with respect to teamwork criteria. Teamwork audits are reported for several aircrew systems, including pilot aids for mission control and management, in order to test the validity of the model, and to establish its sensitivity and diagnostic power. Conclusions are drawn about interface design requirements affecting pilot-aiding and mission performance.

2. INTRODUCTION

Human-Electronic Crew teamwork is the co-ordination of activities of human and machine components of advanced crew systems, employing both conventional and artificial intelligence computational techniques, where there is an orientation towards common goals and objectives. Paradigms and metaphors, such as Human-Electronic Crew teamwork, provide both guiding and limiting frameworks for structuring thinking about crew systems interface design. Traditionally, aircrew interface design has been guided by the manual control paradigm, involving a closed-loop negative feedback control system, with the human as the adaptive element. Improvements in flight technology and aircraft capability revealed limitations on manual control in IMC/IFR conditions, with the potential for loss of control in highly dynamic environments, unusual positions and high G. The introduction of automation technology gave currency to the supervisory control paradigm, with the transfer of some human functionality to automation, and with the human operator allocated a system management role. Experience has revealed the unreliable nature of human monitoring performance, with problems of undetected degradation, and reduced operator intervention capability. Now, the prospect of utilising machine or artificial intelligence (AI) has encouraged the use of the problem-solving paradigm for interface design. This focuses design effort on how the interface can be created to help perform the mission (1).

In 1988 and 1990, participants in two Joint GAF/ RAF/USAF Workshops on the Human Electronic Crew discussed evidence, and broadly agreed that teamwork provides an appropriate metaphor for characterising the relationship between the human and AI system components needed to solve mission problems (2,3). In the military aviation environment, mission problems are characterised by uncertainty and ill-structure. They require flexibility in handling contingencies as they arise, rather than as planned. In dealing with this requirement, AI system interface design needs to reduce operator workload, improve operator situational awareness, and enhance decision-making performance by creating an improved integration or matching between the human and electronic crew capabilities. The difficulties that seem most likely to arise are in the areas of creating trust, maintaining goals and in achieving appropriate levels of autonomy in such a teamwork relationship.

The aim of this paper is to develop an improved understanding of the requirements for teamwork in the Human-Electronic Crew with reference to the literature on social psychology and computer aiding. Through this analysis, it is intended to identify key dimensions that characterise levels of maturity in teamwork, with particular regard to problem solving and decision making under uncertainty. We will describe how these dimensions can be incorporated into a prototype audit tool for evaluating the quality and maturity of Human-Electronic Crew teamwork. We will then test the validity of the model by applying the audit tool to systems intended to aid the pilot including aids for mission control and management.

3. TEAMWORK MODEL

In order to examine the requirements for Human-Electronic Crew teamwork, we will be guided by a generic model representing the system of relationships between different aspects of teamwork. The model, initially proposed at the 2nd Workshop on the Human Electronic Crew (4), is shown in Fig. 1. This teamwork model is derived from the social psychology of small group dynamics (5). Teams differ from small groups in the greater emphasis placed in teams on clear definition of goals, roles and structure. Teams have three distinctive characteristics:

a. Co-ordination of activity, aimed at performing certain tasks and at achieving specific, agreed goals.

b. Well-defined organisation and structure, with members occupying specific roles with associated power, authority and status, whilst exhibiting conformity and commitment to team norms and goals.

c. Communication and interaction between team members, which we refer to as team processes.

The system of relationships between the components of teamwork can be understood in terms of the team's goals, resources, structures, and processes, and their effects on individual team members, team development and team performance. Two system feedback loops can be identified, namely:

a. Feedback on performance of the team's task compared with team goals, possibly leading to changes in team resources, e.g. recruitment of additional expertise.

b. Feedback affecting team structure as both individual members and the team develop, learn and adapt to changing goals and task demands, e.g. dynamic function allocation, initiative turn-taking, emergent leadership.

Requirements for Human-electronic Crew teamwork can be examined in relation to the individual components of this generic model. In this analysis, each component is addressed in two parts. Firstly, we present a summary of relevant heuristics and guidance on teamwork derived from a selective review of social psychology literature (6, 7, 8, 9). Secondly, we identify some of the principal issues raised for Human-Electronic Crew teamwork, based on current applications of decision support systems, with relevance to the teamwork model components.

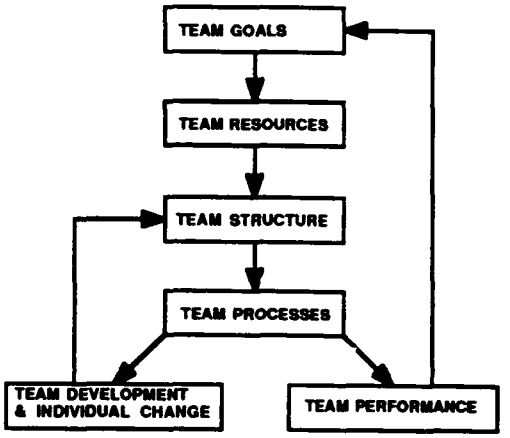


FIGURE 1 - Teamwork Model

4. TEAM GOALS

4.1 Team Goals: Social Heuristics

Effective teamwork is strongly associated with a clear understanding by all members of the team's performance objectives. Effective teams are characterised by a commitment, focus and concentration on clearly defined goals. Team goals should be believed to be worthwhile, and personally challenging. Their pursuit should create a sense of urgency and progress. Achievement of team objectives should make a clear difference to the situation. Failure of teamwork is most commonly caused by the elevation of other goals, usually personal or political, above the team's performance objectives. Personal and political agenda threaten the clarity of team goals, leading to loss of focus and reduced concentration of team efforts. Achievement of objectives can be facilitated by the setting of high, but achievable, performance standards, which motivate and produce pressure on team members to

improve both individual and team performance.

4.2 Team Goals: Human-Electronic Crew Issues

When considering the design of a goal-based team structure, it is necessary to make the distinction between goals, sub-goals, and meta-goals. Goals are the teams objective, successful attainment of which are both necessary and sufficient for task completion. Sub-goals refer to the lesser objectives, attainment of which are necessary but not sufficient for task completion. Sub-goals are relevant where achievement of mission goals requires a staged or iterative process, with the sub-goals being the objectives of each stage (10). Meta-goals refer to overriding goals which, although not being directly related to the task, provide an infrastructure in which successful goal achievement can occur. An example of a meta-goal is the maintenance of pilot situational awareness (11). Although it is not part of the mission, per se, it is an important factor in the pilot's ability to reach task goals. Failure to achieve such a meta-goal will impact on task performance. Thus meta-goals are most relevant during the design stage, be it the design of systems or team structures.

When team structures are functioning in a dynamic environment, then sub-goals (and to a lesser extent task goals) will change (12). For effective team performance, all team members must be aware of any change in their goals. Failure to communicate such changes will result in the loss of the common goal structure. The impact of this for the design of human-computer teams is that such communications must be bidirectional, i.e. sufficient feedback must be given by both team members for the other to maintain his awareness of the new goals.

5. TEAM RESOURCES

5.1 Team Resources: Social Heuristics

Team resources comprise all the relevant abilities and tools, skills, rules and knowledge available to perform the task, including both human and machine capabilities. The availability of resources is determined by the team size, i.e. the number of individual members, by the level of individual and team training, competence and expertise, and by situational factors such as fatigue, boredom and anxiety stress. Increasing team size may facilitate or inhibit performance on tasks, e.g. by generating more ideas on problem solving tasks, whilst increasing the time taken to generate each idea. Resources may be redundant or unique, through the team being composed of homogeneous or heterogeneous membership. Homogeniety for some resources may be more effective for performance than heterogeneity. However, resource variability, through heterogeneity may produce greater sensitivity to changing task demands. The resources must be willing and able to collaborate effectively. Compatibility can be achieved with different but complementary resources for dimensions such as the need to dominate or to control events. Teamwork is normally associated with the achievement of specific goals requiring specialised skills. For effective teamwork, the requisite resources should be available and appropriately distributed among the team members, in accordance with the task structure and load. In tasks where the performance of a team is limited by the weakest member (conjunctive task constraints), resource variability is undesirable. In tasks where team performance is determined completely by the best member's performance (disjunctive task constraints), a high degree of resource variability is desirable. Matching of resource characteristics to task demands, in terms of the difficulty of underlying operations, is the key to estimating the capability for effective teamwork.

5.2 Team Resources: Human-Electronic Crew Issues

The design of the human-computer team requires that knowledge be gained of the resources of each team member. This will allow the a priori allocation of tasks to that member best fitted to perform them (where expertise is limited to one member) or dynamically according to the situational demands (where both members have relevant expertise). Correct utilisation of unique and redundant resources will produce a synergistic team, thus extending the total resources of the team (13). An example where unique resource allocation would be effective is in judgments under uncertainty. Humans are traditionally poor at integrating multiple sources of probabilistic information in a formal statistical manner (14). Computers are, on the other hand, good at such 'number crunching' activities. Thus a successful team would use the computer resource to achieve this part of the process. Humans are good, however, at accepting integrated advice and using heuristics to make judgments under uncertainty (15). Thus the team would use this human resource to complete the decision process.

Where both team members have the expertise and resources to perform a function, then allocation of that function should be performed dynamically, with the choice of who should do the task being decided through consideration of meta-goals, e.g. maintenance of SA, reduction of pilot workload etc.

Another consideration in the integration of human and computer resources to provide enhanced goal attainment, is in the method of representation of task information by each party, such that a synergistic relationship can be attained. Triggs (16) showed

that the ability of human decision makers to integrate multiple sources of probabilistic information was dependent, not just on the difficulty of the task, but also on the method of information representation used in the task. Further, Green (17) claims the knowledge structures of experts and novices differ in their organisation and hence the utilisation of such resources within a human-electronic crew will require that such differences be taken into account by system designers. Thus, the design of suitable task structures to exploit the available resources will be critical in for the successful integration of human-electronic crew teams. Some issues relevant to the design of these systems are discussed below.

Operator capability analysis is a key technology for matching human resource capabilities to mission performance objectives (18). This analysis needs to be extended to encompass both human and electronic crew capabilities. A common performance resource model and associated taxonomy is required for systematically linking resource capabilities to task demands, including engineering, aptitude and training parameters of both the human and electronic crew team components.

6. TEAM STRUCTURE

6.1 Team Structure: Social Heuristics

Team structure concerns the relatively stable pattern of relationships between members that determines the communication required for co-ordination of activities, and that governs the distribution of functions, roles, status and power. The function of team structure is to implement access to task-relevant resources. Team structure and associated patterns of communications should be designed to facilitate rather than restrict access. Effective teamwork requires a structure driven by performance results. The required structure is that which is appropriate for achievement of the specific team performance objectives, with the minimum complexity of resources necessary and sufficient for successful functioning. Maintenance of organisational processes should not consume unnecessary resources. In a functionally effective structure, individual and team efforts always lead towards achievement of the team goal. Increasing cohesiveness and attraction between team participants leads to greater conformity to team norms, more uniformity of behaviour, improved performance and increased job satisfaction. Poor performance and morale are associated with poor cohesiveness and low membership attractiveness. Centralisation of communication structure affects membership satisfaction and team performance.

More centralised communication networks (e.g. wheel versus circle structure) produce better team performance but lower membership satisfaction, except in complex tasks when the central elements become overloaded. Decentralised networks allow a more even distribution of workload (Fig. 2).

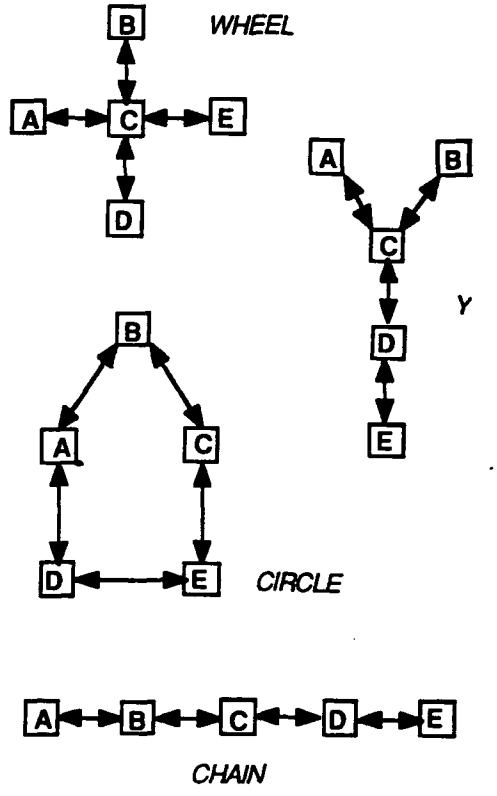


FIGURE 2 - Team Communication Structures

Structure creates role differentiation. Status and power are affected by roles and functions. Discrepancy between the expected role, perceived role and enacted role of individuals introduces conflict between team members. The evaluation attached to a role determines the status of the individual and influences conformity to team norms. The perception that interactants have equal status facilitates communication. Function allocation is essential for effective co-ordination of goal-oriented activities. A high degree of rigidity and clarity in function allocation, with clear accountabilities, is beneficial for well-structured tasks, which follow a clearly defined plan, and for tasks requiring unique rather than redundant resources. Flexibility in function allocation is beneficial for tasks involving ill-structured problems, uncertainty and requiring good communication between team members. The communication structure influences the distribution

of information, power and authority in teams. The member through which most information passes has the potential to exert considerable influence over the team (informational power). In a highly-centralised communication structure, the member in the central role acts as the information gate-keeper. This member is most likely to be perceived as, and act as the team leader. The distribution and locus of power depends on their ability to monitor the performance of members and to exercise reward, coercion and feedback (reward and coercive power), to generate a positive image that attracts emulation (referent power) and by the ability to internalise goal-relevant information and acquire knowledge (expert power). Effective teamwork is most likely to occur when assigned, legitimate power coincides with the locus of informational, expert, referent, reward and coercive power. The function of the team leader is to exercise authority and power in order to achieve the team's performance objective. Leadership is achieved by changing, directing and controlling the behaviour, attitudes and opinions of team members to conform with team roles, standards, norms, and goals. Leadership behaviour includes clarifying team objectives, making important decisions and taking initiatives, and identifying ways of achieving objectives. Leadership effectiveness is dependent on the exercising of situationally appropriate task-oriented and relationship-oriented skills. A strongly authoritarian, task-oriented style is not necessarily the most productive when dealing with ill-structured tasks requiring good communication and cohesiveness between members.

6.2 Team Structure: Human-Electronic Crew Issues

The implementation of the structures described above in the human-computer team require consideration of the levels of autonomy to be assigned to the computer's functions. Figure 3 shows a graphic conceptualisation of two alternative team structures, with the electronic crew member in a subordinate (tandem) and in associate (side-by-side) roles. Several taxonomies of the levels of autonomy of human-computer interaction have been postulated (19,20,21). Sheridan et al (19) describe the level of interaction as varying from the human performing all functions, through increasing levels of computer autonomy, to the computer performing functions independently with discretionary power to not inform the human. As the computer assumes more responsibility, then its role changes. They characterise these roles as 'tool', 'assistant', and 'autonomous agent' with respect to their increasing autonomy. Although it may be argued that more levels could be included in the model, it does provide a useful metric against which to measure the maturity of currently available systems, as well as an a priori design aid in the development of such systems. In other words, such a taxonomy allows you to choose the level of interaction that you judge to be more relevant for a particular task, and then to judge existing systems against that to see how closely the design aim has been approached. The level chosen is likely to be situationally dependent in that it will depend on the demands of both goals and meta-goals. This is since the requirement for autonomy is itself a dynamic function of the operational context.

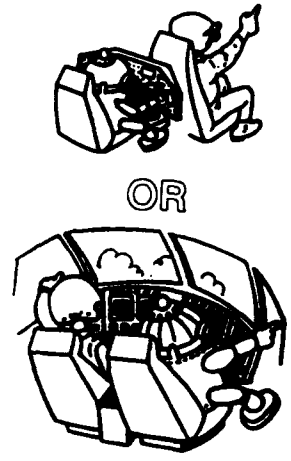


FIGURE 3 - Human Electronic Crew Structures

For efficient teamwork, the locus of power should be allocated to maximise goal achievement. An example would be the computer taking control when the pilot suffers G-induced loss of consciousness. Also, where the computer controls data fusion and displays management, e.g. Pilot's Associate, and hence information flow, then this necessitates, by implied consent, that at least part of the leadership role (traditionally associated with the human team member) will be performed by the computer team member. A limit on the autonomy of the machine element of the team is also likely to be imposed by the available knowledge state of the machine. Zachary et al (22) classify decision situations according to how many of the following elements of the decision are known.

a. Goals: desired states against which outcomes are matched.

b. Knowledge: representation of relevant real-world information.

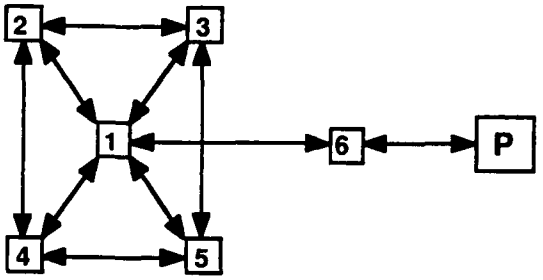
c. Action Options: legal operators that can be used to alter the situation.

d. Action Outcomes: states resulting from action options.

e. Desirability Functions: how well do outcomes satisfy goals.

They argue that full automation of decisions is only appropriate when all or nearly all of these elements are known. When knowledge is lacking, i.e. uncertainty exists, then supporting the human decision maker is relevant.

Figure 4 shows the communication structure proposed for the six major sub-systems comprising the Pilot's Associate (23). The executive sub-system was originally conceived as an overall executive manager, residing between the individual sub-system experts with responsibility for co-ordinating their actions. Subsequently, doubts have arisen over the need for this separate executive function, in addition



PA Modules Key:

1. Mission Executive/Manager
2. Situation Assessment
3. Tactics Planner
4. Systems Status
5. Mission Planner
6. Pilot Vehicle Interface

P. Pilot

FIGURE 4 - Pilot Associate Communication Structure

to the Pilot Vehicle Interface (PVI) manager expert (3). Deletion of this separate executive function could equate to the loss of a redundant layer of bureaucracy in order to facilitate access to primary resources.

Where task allocation is static, the level of interaction can be chosen a priori. Dynamic task allocation may require a variable authority gradient to exist in order to best exploit the resource distribution across team members. Where situational demands are high, requiring pilot mandate for all low level 'chores' may decrease the usefulness of assigning such tasks to the computer. When the demand is less, then less autonomy may be beneficial by maximising the degree to which the pilot is 'in the loop'. The extent to which this is relevant is likely to depend on the types of functions which the computer can perform, and also on the degree of trust in the computer exhibited by the human. Where a high degree of trust is available, task leadership functions may be shared or even transferred to the computer member. How such levels of trust can be produced are discussed in the next section. Thus the level of interaction chosen, and hence the most appropriate team structure, will be dependent on the task being performed, knowledge of the task elements available to each part of the team, the requirement for workload reduction/SA maintenance, the humans trust in the machine's ability, the need to keep the pilot 'in the loop', the efficiency of the man-machine communication, and knowledge of what the other member is doing.

7. TEAM PROCESSES

7.1 Team Processes: Social Heuristics

Team processes of communication and interaction are affected by the structural characteristics of the team (roles, status, cohesiveness). Communications can be analysed for functional content and style. Content can be task oriented or social-emotional oriented. Task-oriented communication includes interactions exchanging information (repetition, confirmation, clarification), opinions (evaluation, analysis, feelings) and direction (suggestions, possible actions). Communication with social-emotional orientation involves positive and negative reactions concerning agreement (acceptance, concurrence, understanding), satisfaction (release of tension, humour) and solidarity (affirmation of status, help, reward). Social cohesiveness is important for team productivity and performance. Interactive styles can be characterised as affiliative-nonaffiliative, affecting cohesiveness and reflecting attractiveness; dominant-submissive, reinforcing power relationship and reflecting status; responsive-unresponsive, reflecting the expressive quality and effectiveness of communication.
Communication has temporal and bandwidth constraints (channels, modalities). Unrestricted communication can be unproductive, distracting and an inefficient utilisation of resources. Broadening the communication bandwidth increases the psychological closeness of interactants. However, some psychological distance may be necessary for

tasks requiring autonomy and independence of thought and action. Widening the bandwidth beyond that needed for audio communication does not improve performance on some problem-solving tasks. The communication bandwidth should be that which is necessary and sufficient for achievement of team goals. Formal language can be restrictive, slow and inefficient for solving ill-structured problems. Conformity to dialogue protocols (e.g. rules for structuring turn-taking, transferring controls) should increase the efficiency of communication and maintain the goal-orientation of interactions. Effective communication requires knowledge of the functionality, meaning and goal of the communication. This is achieved by tracking both the goal and the context of the communication, using domain-specific information and information for the control of the communication process.

Effectively communicating and collaborating teams are characterised by a high degree of trust. Trust requires predictability, dependability and faith in interpersonal relationships. Trust occurs in a collaborative climate characterised by honesty, openness, consistency and respect. Violations of trust have catastrophic, irredeemable effects on team functioning and performance. Full teamwork potential is unlikely to be realised when trust has been broken. Trust allows team members to stay problem-focused, it promotes efficient communication and co-ordination, improves the quality of collaborative outcomes, and it leads to compensatory behaviour. Compensation between individuals is necessary for performance standards to be independent of variability in team resources.

7.2 Team Processes: Human-Electronic Crew Issues

Communication between the human and computer team members is achieved through the design of the interface between them. An effective interface requires an understanding of the knowledge requirements of the team members with a consequent moding of input/output facilities to support these requirements. Again, both goals and meta-goals need to be considered in the design of the interface. If the computer is to control the flow of information, then an efficient model of the human's information processing abilities/ requirements is essential. Adaptive or learning interfaces have the potential for maximising teamwork (24). The problem with such systems in that they can learn 'bad habits' (or sub-optimal behaviours) from the human if they are adapting to his behaviour without sufficient reference to the task and meta-goals. They need to be goal rather than behaviour driven, implying the need for the adaptivity to be bidirectional. In other words, the team efficiency will only be improved where human and computer are given sufficient feedback on goal achievement to both learn, and hence improve sub-optimal performance.

An alternative approach is to embed in the system sufficient knowledge to allow it to make a priori judgments of the operator's knowledge requirements and to structure the interface accordingly. An example of such an interface is the Pilot Vehicle Interface (PVI) described by Shelnutt (25, 26). The PVI employs an expert system containing knowledge about goals, actions available etc, and supports the integration of the operator with both his expert and non-expert systems by providing dynamic sensor- and information-fused displays. The embedded knowledge it contains allows it to mode these displays to be relevant to the task being performed by the operator, thus giving him the information he needs when he needs it. The advantage of this type of approach is that it ensures that common goals and knowledge exist between human and machine, and that such goals may be maintained even in dynamic situation.

Such goal tracking and commonality will not only allow compensatory team work to occur, but will also enable a suitable level of trust to develop and be maintained. Failure to provide it may result in over-trusting (27) or under-trusting (28,29), both of which impact negatively on task goal achievement (30). Eimer investigated the effectiveness of a decision support system (DSS) in a command, control, and communications task involving uncertainty. He simulated decision support (which could be either of human or machine origin) of varying accuracy (i.e. how often their advice was correct) to examine the relationship between operator performance and DSS performance. Further, he investigated the effect on operator performance of the system breaking down, i.e. providing invalid advice continuously. He found that operator performance decreased as the effectiveness of the DSS decreased. Where DSS accuracy was low, then it produced worse performance in subjects than the control group with no DSS. When a previously valid DSS became invalid, then performance again became worse than that of the control group. In other words, overtrust in an inaccurate system, or one which ceases to function, interferes with the task and produces worse performance than no system at all.

A lack of trust is liable to result in a "boy who cried wolf" situation, where teamwork will break down due to a judgment of machine generated information as being unreliable and hence untrustworthy. This will result in either the pilot ignoring the information or seeking to check it for himself. Either of these outcomes is liable to produce sub-optimal teamwork. This is supported by an experimental investigation

into the effect of trust on a discussion support system reported by Lerch and Prietula (28). They measured subjects agreement and confidence in human and expert system generated advice. They found that initially trust increased rapidly as correct advice was given. Following a trial where incorrect advice was given, however, confidence was significantly decreased on future trials and failed to return to the level achieved before the incorrect advice trial. Thus it seems likely that any system with low accuracy is likely to be considered as untrustworthy by the human, and hence the interaction between them will be degraded.

Taylor (31) attempted to quantify the trust required between a human-machine crew by measuring the factors affecting trust between two-man human teams (pilots and navigators). He differentiated between factors affecting the demand for trust and those affecting the supply of trust. Unless the two are matched, then optimum team co-operation will not be achieved. Each, he says, can be manipulated to improve trust within the team. Demand for trust can be reduced by minimising the risk and negative payoffs associated with incorrect decisions, and also by ensuring that a common goal/intent structure exists and is consistently applied. The supply of trust can be improved by the machine reducing the uncertainty associated with decisions thus increasing the probability of success in them and hence the pilot's confidence in those decisions. Further, by representing that uncertainty to the pilot (rather than removing it), it provides a degree of transparency to the system, allowing the human to understand, and hence trust the machine advice more readily.

8. TEAMWORK MATURITY AUDIT

8.1 Audit Objectives

On the basis of the forgoing analysis, assuming that the teamwork model is valid, it should be possible to create tools for evaluating and auditing the quality and maturity of teamwork in candidate Human-Electronic Crew systems. Information gained from audits could serve to guide future developments aimed at improving system teamwork performance. Audit information could also serve to assess the validity of the teamwork model and to test the model's sensitivity and diagnostic power.

8.2 Audit Constructs

A selection of possible audit constructs, associated with teamwork maturity, linked to the principal model components, is shown in Table 1. Additional constructs from other domains, e.g. systems architecture, software engineering etc will need to be included in a fully comprehensive analysis. The level of teamwork maturity in Human-Electronic Crews could be assessed by establishing the strength of teamwork constructs in the candidate systems, judged by appropriate experts, such as system designers, or preferably, system users. A simplified method might establish the extent to which the constructs are judged to be, say, primary features, minor features, or not represented in the system. Alternatively, the strength of the constructs could be measured using questionnaires, rating scales, repertory grids, or other subjective measurement techniques. A test of the validity of the teamwork model could be achieved by measuring the extent to which the derivative audit tools are sensitive to changes in pilot-aiding systems using old (immature) and new (relatively mature) technology, and that they can discriminate the causes of differences in mission performance.

8.3 Audit Candidates

In order to provide some preliminary worked examples for discussion, and so as to address some of the validity issues, sample teamwork audits have been conducted, using the prototype audit constructs, on eight candidate systems, designed for different operational roles, with both immature and advanced technology. Short technical descriptions of the audited systems are provided below.

8.3.1. Civil Transport Role

a. Immature Candidate - Piper Apache PA28/7

> Mid-1950's, twin piston mono-plane, with retractable gear, constant speed propellers, split flaps, full anti-icing kit; a nominal 4-seater with dual-controls and standard blind flying panel, designed for single-pilot operation, used for instrument training and air taxi.

b. Mature Candidate - A320 AIRBUS

> Mid 1980's, advanced state-of-the-art twin jet, medium range passenger aircraft, designed for 2-pilot operation, with glass cockpit, fly-by-wire and side-stick controllers, full FMS, integrated instrument presentation, auto throttle and auto pilot. FMS is loaded by computer disc and instrument specialist.

MATURITY CONSTRUCTS	DEFINITIONS
TEAM GOAL	
Clarity	Clearly defined performance objectives.
Common Structure.	Shared understanding of meta/sub goals.
Tracking	Awareness of changing objectives.
Impact	Critical for mission success.
Achievement	High probability of success.
TEAM RESOURCES	
Sufficiency	Enough expertise/ability/competence.
Availability	Readiness for application to task.
Heterogeneity	Variability/uniqueness of expertise.
Compatibility	Ability to combine/integrate/match.
Enhancement Capability	Ability to add to expertise.
TEAM STRUCTURE	
Goal Driven	Governed by performance objectives.
Resource Accessibility	Facilitates access to resources.
Cohesiveness	Attracts conformity to team norms.
Dynamic Function Allocation	Real-time role-task distribution.
Levels of Autonomy	Degrees of independent functioning.
TEAM PROCESSES	
Wide Bandwidth	Multiple modalities for communication.
Bidirectionality	Two-way flow of information/feedback.
Shared Initiative	Leadership turn taking.
Common Knowledge Base	Shared understanding of situations.
Trust	Willing to accept others' judgments.

TABLE 1 - Prototype Teamwork Audit Tool

8.3.2 Air Defence Role

a. Immature Candidate - BAe Hawk

Mid 1970's, two-seat tandem, rugged, agile jet, with commercial cockpit instruments and flight controls, designed as an advanced, tactical weapons trainer for fast-jet pilots, and employed in a secondary air defence war role.

b. Mature Candidate - General Dynamics F16C Fighting Falcon

Late 1970's multi-role single-seat, high manoeuvrability, light-weight fighter jet, updated with advanced air-to-air combat and air-to-ground capabilities, with fly-by-wire side-stick controller, advanced digital avionics, radar, and weapons delivery systems, low altitude, Infra-red LANTIRN night attack, automatic electronic jammer, threat analysis and voice warning system.

8.3.3 Strike Attack Role

a. Immature Candidate - Panavia Tornado GR1

Late 1970's, variable geometry (swing wing), two-seat tandem, multi-role jet, employed in the overland strike/attack and reconnaissance roles, with all-weather, night automatic Terrain Following, Inertial and Doppler navigation radar; with pilot E-scope and moving map display, automatic laser/radar or laser/HUD weapon aiming and delivery, ECM radar warning; and with navigator combined Radar and Projected Map Display, Electronic digital TV tabular displays for mission computer monitoring and planning, and with mission plan pre-loading facility.

b. Mature Candidate - Mission Management Aid (MMA) UK MOD/Industry Joint Venture

Collaborative research project aimed at establishing functional requirements and prototyping a system to aid single-seat jet mission management in Air-to-Ground role, with future Air-to-Air enhancement capability.

Sensor Fusion, Situation Assessment, Dynamic Planning, and Man-Machine Interface core functions generating a tactical plan for pilot acceptance/rejection, presented through the pilot controls/displays, including current situation information, proposed MMA action/solutions/explanations, systems status and cueing, and automatic information prioritisation/scheduling compatible with the pilots current tasks and mission objectives.

8.3.4 Ground Planning Role

a. Immature Candidate - Jaguar Mk 1 Aircraft, Ferranti Autoplan

Mid 1970's digital aircraft route planning system for the Jaguar Mk 1 tactical strike and ground attack aircraft, for use in conjunction with standard air charts, comprising digitised chart plotting table, cursor for positional data inputs, electronics unit, paper tape printer and control panel, with outputs of track, fuel, time etc to 25 metres accuracy on a 1:50K chart, and with a Portable Data Store for automatic transfer to the aircraft's digital navigational system.

b. Mature Candidate - Harrier GR Mk 7 Aircraft Advanced Mission Planning Aid (AMPA)

Currently under UK MOD procurement, an Advanced Mission Planning Aid (AMPA) to provide computer assistance for ground mission planning of Harrier GR Mk 7 close-air support, interdiction, counter air and recce operations and training. Employing advanced, state-of-art networked, computer-based mission planning workstations using from raster scanned, and eventually digital vector map data, with high resolution colour map image display, overlays, panning, zooming and pointing facilities, full graphics package, manual and automatic route planning, calculation and editing aids, 3-D route visualisation simulation and intervisibility plan, grey scale image handling, and hard copy and electronic aircraft outputs.

8.4 Audit Method

Understanding and communication of the meaning of the teamwork audit constructs are likely to be sources of variation in audit results. To minimise these principal sources of variation, and to simplify the task in hand, a single system expert was sought for each of the four roles. This was achieved, with one exception. In the air defence role, two experts co-operated to provide a joint assessment. The Civil Transport Role audit was provided by Capt Paul Wilson, Head of CHIRP Group, Psychology Division, RAF IAM, Civil Pilot and Aviation Consultant to the CAA. The Air Defence Role audit was provided by Lt Col (USAF) Geoff McCarthy and Sqn Ldr Terry Adcock, RAF IAM Medical Officer Pilot and Test Pilot, respectively. The Strike/Attack Role audit was provided by Sqn Ldr Phil Price, OR 52c(Air) MOD, a Tornado Pilot, currently responsible for advanced cockpit operational requirements. The Ground Planning Role audit was provided by Sqn Ldr J.W.J. (Ted) Taylor AES 11, MOD(PE) , an RAF Navigator currently responsible for the procurement of mission planning systems.

These 5 experts provided the audit data for both the immature and mature candidates within their respective roles. Following briefing on the purpose and scope of the teamwork model and task, and two candidate system were evaluated with regard to the 20 teamwork constructs, based on a common, agreed understanding of the construct meaning.

In each case, the expert(s) were required to decide whether each audit construct was a primary feature (Score = 2), a minor feature (Score 1) or not, represented in the system (Score = 0).

8.5 Audit Results

The results of the audit are summarised in Table 2 and displayed graphically in Fig 5. There is insufficient data points to conduct a reliable, meaningful statistical analysis. The maximum possible score is 10 for each of the model components, and 40 overall. The assigned scores ranged widely from 1/10 (Autoplan: Processes) to 10/10 (A320: Goals; GR 1: Goals, Processes) within components, and from 10/40 (Autoplan) to 36/40 (GR 1) across components. With the exception of the Strike/Attack Role, the hypothesised "Immature" Candidates scored less in total and less within components (N.B. one equal) that the comparable Mature Candidates. The Tornado GR 1, was assigned the highest score (36/40), and exceeded the score assigned to the MMA (22/40), running counter to the a priori maturity hypothesis.

Comparing across the candidate system, the lowest scores were assigned to the Team Processes (40/120 and the highest to team Goals (53/120). Comparing the individual constructs, total scores ranged from 5/16 (Goal Tracking) to 15/16 (Heterogeneity).

TEAMWORK MATURITY CONSTRUCTS	TRANSPORT		AIR DEF		STR/ATT		PLANNING		T
	PA28	A320	HAWK	F16C	GR1	MMA	AUTO	AMPA	
GOALS									
Clarity	0	2	0	2	2	2	2	2	12
Common Structure	0	2	0	2	2	1	1	2	10
Tracking	0	2	0	0	2	0	0	1	5
Impact	2	2	2	2	2	0	1	2	13
Achievement	2	2	2	2	2	0	1	2	13
SUB TOTAL	4	10	4	8	10	3	5	9	53
RESOURCES									
Sufficiency	1	2	1	2	2	1	0	2	11
Availability	1	1	1	2	2	1	1	1	10
Heterogeneity	2	2	2	2	2	2	1	2	15
Compatibility	0	2	0	2	1	1	0	2	8
Enhancement	1	2	0	0	1	2	0	2	8
SUB TOTAL	5	9	4	8	8	7	2	9	52
STRUCTURE									
Goal Driven	2	2	2	1	2	1	1	2	13
Accessibility	2	1	2	2	1	2	1	2	13
Cohesiveness	0	2	2	2	1	1	0	2	10
D.F.A.	0	2	0	0	2	2	0	0	6
L.O.A.	0	2	0	1	2	2	0	2	9
SUB TOTAL	4	9	6	6	8	8	2	8	51
PROCESSES									
Wide Bandwidth	1	1	1	1	2	1	0	1	8
Bidirectionality	1	1	0	1	2	1	0	1	7
Shared Initiative	0	2	1	0	2	0	0	1	6
Common Knowledge	1	1	1	1	2	1	0	1	8
Trust	1	2	1	2	2	1	1	1	11
SUB TOTAL	4	7	4	5	10	4	1	5	40
GRAND TOTAL	17	35	18	27	36	22	10	31	196

TABLE 2 - Teamwork Audit Results

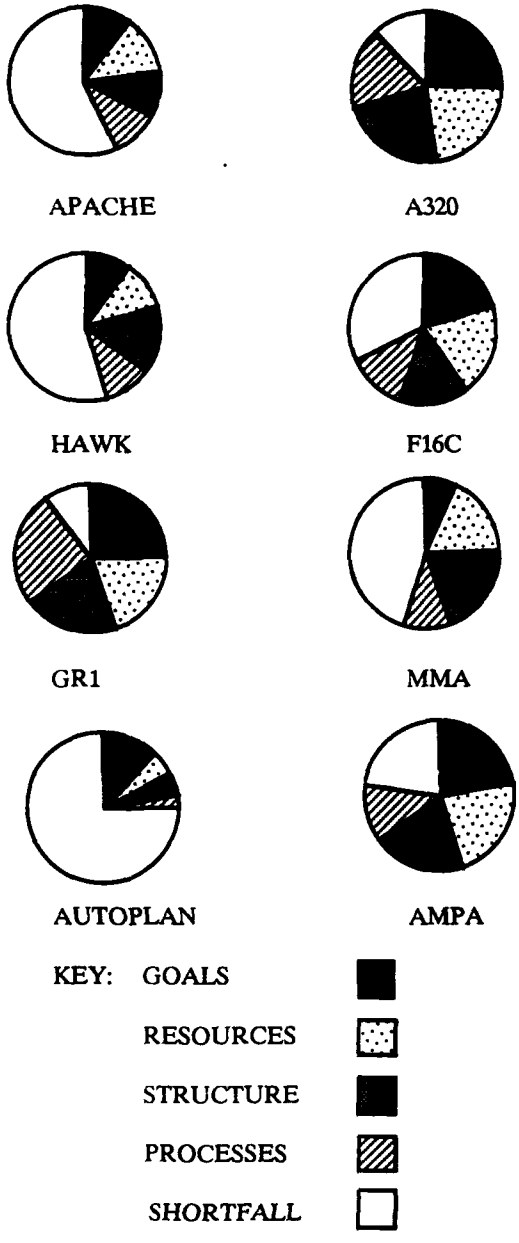


FIGURE 5 - Audit Component Proportions

8.6 Audit Discussion

The audit data provide the briefest, outline sketch of the pattern of teamwork across crew-system roles and technologies. The absence of statistical procedures makes it impossible to draw firm conclusions. However, in general the data seem to support the notion that the teamwork model is at least sensitive to the substantial developments in crew systems technologies that have occurred since the early 1970s. They lend support to the observation that advanced computer technology has enabled widespread embodiment of mission goal requirements in crew-system design and enhanced mission control and management. On the other hand, the data show little evidence of substantial improvements in teamwork processes, supporting the common observation that MMI developments tend to lag behind progress in mission-system capability. Notwithstanding, advanced teamwork concepts, such as dynamic function allocation (DFA) and levels of autonomy (LOA), probably are significant, substantive developments in teamwork structure.

Comparisons between assessments across roles are confounded by the choice of system and auditor for comparison. Both the MMA and AMPA are future systems under development. Consequently, their assessments are necessarily speculative, and therefore less reliable than those concerned with operational systems. It should be noted that the comparatively optimistic AMPA assessment was provided by a member of the project management team, whereas the more pessimistic MMA assessment was provided by an informed, but healthily sceptical Tornado pilot. The Tornado pilot's brief was to compare his understanding of the prototypical MMA with the current operationally successful mission system, coupled with the Navigator. Hence, an unusually high evaluation is given of the GR 1 team processes, which are strongly based on successful human-human communication. An audit by the MMA design team probably would have been more optimistic since many of the model features are compatible with MMA design goals. Although these limitations on the audit technique undoubtedly exist, and limit the conclusions that can be drawn from the data, this approach to systems assessment probably has potential for further development as a high level diagnostic tool, if supported by more scientific, systematic data gathering procedures.

9. CONCLUSIONS

The proposition that teamwork should be a high-level design driver poses a radical departure from conventional systems engineering objectives. This review demonstrates that there is a substantial understanding, in social psychology, of the processes of teamwork, sufficient to generate a potentially extensive list of criteria for judging the quality and maturity of Human-Electronic Crew teamwork. The limitations of the model are that Human-Electronic Crew teamwork may have unique emergent characteristics that are not evident from analysis of human teamwork. A major problem with the model is that it is based on a high degree of trust. When distrust occurs, teamwork breaks down in a potentially catastrophic and irredeemable manner. This may result, in the worst case, in the human

operator refusing to use any of the electronic crewmember's capabilities. The distribution of power raises the issue of leadership, with consequent moral and political implications if the locus of power is not to reside with the human. Such a prototype model audit is unlikely to provide a fully comprehensive analysis of the issues. However, the aim of audit is to judge the whole through a sample of relevant criteria. Although more statistically testable validation is required, this model may have some utility, at least, in conceptualising, designing, and validating candidate Human-Electronic Crew teams. The provisional results reported here indicate that the model is sufficiently relevant and understandable to be applied by system users with some sensitivity and diagnostic power. It remains to be seen if the model is capable of generating interesting predictions and hypotheses that can be tested both empirically and operationally.

10. REFERENCES

1. Egglestone, R.G., "Machine Intelligence and Crew-Vehicle Interfaces, *in* E. Herr and H. Lum (Eds.) "Machine Intelligence and Autonomy for Aerospace Systems", AIAA, 1988

2. Emerson, T.J., Reinecke, M., Reising, J.M. and Taylor, R.M. (Eds.) "The Human-Electronic Crew: Can They Work Together?" Proceedings of the 1st GAF/RAF/USAF Workshop, BSD-DR-G4, December 1988. RAF Institute of Aviation Medicine, Farnborough, Hants, December 1988. Tech Report WRDC-TR-89-7008 Wright Patterson AFB, OH: Cockpit Integration Directorate, 1989.

3. Emerson, T.J., Reinecke, M., Reising, J.M. and Taylor, R.M., "The Human-Electronic Crew: Is the Team Maturing?" Proceedings of the 2nd GAF/RAF/USAF Workshop, PD-DR-P5. RAF Institute of Aviation Medicine, Farnborough, Hants, April 1991.

4. Selcon, S.J. and Taylor, R.M., "Psychological Principles for Human-Electronic Crew Teamwork", *in* Proceedings of the 2nd Workshop on the Human Electronic Crew. PD-DR-P5. RAF Institute of Aviation Medicine, Farnborough, Hants, April 1991.

5. McGrath, J.E., "Social Psychology: A Brief Introduction", New York: Holt, Rinehart & Winston, 1964.

6. Pennington, D.C., "Essential Social Psychology", London: Arnold, 1986.

7. Eimer, E.O., "Team Problem Solving Effects of Communication and Function Overlap", AAMRL-TR-87-037. AFSC AAMRL WPAFB: Dayton, OH. 1987a.

8. Wellens, A.R. and McNeese, M.D., "A Research Agenda for the Social Psychology of Intelligent Machines", NAECON Conference Proceedings, IEEE., 1987.

9. Larson, C.E. and La Fasto, F.M.J., "Teamwork: What Must Go Right/What Can Go Wrong", London: Sage, 1989

10. Simon, H.A., "Information Processing Theories of Human Problem Solving", *in* W.K. Ester, (Ed.) "Handbook of Learning and Cognitive Processes", London: Erlbaum, 1978.

11. Selcon, S.J. and Taylor, R.M. "Decision Support and Situational Awareness", *in* Y. Queinnec and F. Daniellou (Eds.) Designing for Everyone, Vol. 1, pp 792-794, 1991.

12. Lind, M., "System Concepts and the Design of Man-Machine Interfaces for Supervisory Control", *in* L. Goodstein, H. Anderson and S. Olsen (Eds.) "Tasks, Errors and Mental Models", London: Taylor & Francis, 1988

13. Moss, R.W., Reising, J.M. and Hudson, N.R., "Automation in the Cockpit: Who's in Charge?" *in* Proceedings of the 3rd SAE Aerospace Behavioural Engineering Technology Conference, Long Beach, California, pp 1-5, 1984.

14. Kahneman, D., Slovic, P. and Tversky, A. (Eds.) "Judgment Under Uncertainty: Heuristics and Biases", New York: Cambridge University Press, 1982.

15. Ebbesen, E.B., Parker, S. and Konecni, V.J. "Laboratory and Field Analyses of Decisions Involving Risk", Journal of Experimental Psychology: Human Perception and Performance, Vol. 3, pp 575-589, 1977.

16. Triggs, T.J. "The Ergonomics of Decision Making in Scale Systems: Information Displays and Expert Knowledge Acquisition', Ergonomics, Vol. 31, No. 5, pp 711-719, 1988.

17. Green, A.J.K., "The Relationship Between Task Representations, Knowledge Structures and Expertise", Paper to BPS Cognitive Psychology Section, 8th Annual Conference, Oxford, September 1991.

18. Taylor, R.M., "Human Operator Capability Analysis for Crew Systems Design". Proceedings of the BPS Occupational Psychology Symposium, Cardiff. Letter Report No 004/91. RAF Institute of Aviation Medicine, Farnborough, Hants. January 1991.

19. Sheridan, T.B. and Verplanck, W.L., "Human and Computer Control of Underseat Teleoperators", (Tech Rpt) MIT: Cambridge, MA, 1978.

20. Krobusek, R.D., Boys, R.M. and Palko, K.D. "Level of Autonomy in a Tactical Electronic Crewmember", *in* Emerson, T.J., Reinecke, M., Reising, J.M. and Taylor, R.M., (Eds.) "The Human-Electronic Crew: Can They Work Together?" Proceedings the 1st GAF/RAF/USAF Workshop, BSD-DR-G4, December 1988. RAF Institute of Aviation Medicine, Farnborough, Hants, December 1988. Tech Report WRDC-TR-89-7008 Wright Patterson AFB, OH: Cockpit Integration Directorate, 1989.

21. Yadrick, R., Judge, C. and Riley, V. "Decision Support and Adaptive Aiding for a Battlefield Interdiction Mission, *in* Emerson, T.J., Reinecke, M., Reising, J.M. and Taylor, R.M. (Eds.) "The Human-Electronic Crew: Is The Team Maturing?" Proceedings of the 2nd GAF/RAF/USAF Workshop, PD-DR-P5, April 1991. RAF Institute of Aviation Medicine, Farnborough, Hants, April 1991.

22. Zachary, W., Wherry, R., Glenn, F. and Hopson, J. "Decision Situation, Decision Processes, and Decision Functions: Towards a Theory-Based Framework for Decision-Aid Design", National Bureau of Standards, Human Factors in Computer Systems Conference, Gathersaburg, MD, 1982.

23. Small, R.L., Lizza, C.S. and Zenyuh, J.P., "The Pilots Associate: Today and Tomorrow", *in* Proceedings of the 1st Workshop on the Human Electronic Crew. BSD-DR-G4. RAF Institute of Aviation Medicine, Farnborough, Hants. December 1988. WRDC-TR-89-7008. Wright-Patterson AFB, OH. Cockpit Integration Directorate.

24. Weisbrod, R.L., David, K.B. and Freedy, A., "Adaptive Utility Assessment in Dynamic Processes: An Experimental Evaluation of Decision Aiding", IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-7, No. 5, pp 377-383, 1977.

25. Shelnutt, J.B., Stenerson, R., Nelson, P. and Marks, P., "Pilot's Associate Demonstration One: A Look Inside", *in* Proceedings of the First Aerospace Applications of Artificial Intelligence Conference, Dayton, Ohio, 1986.

26. Shellnut, J.B., "Pilot Vehicle Interface Management", *in* Emerson, T.J., Reinecke, M., Reising, J.M. and Taylor, R.M., (Eds.) "The Human-Electronic Crew: Can They Work Together?" Proceedings the 1st GAF/RAF/USAF Workshop, BSD-DR-G4, RAF Institute of Aviation Medicine, Farnborough, Hants, U.K. December 1988. Tech Report WRDC-TR-89-7008 Wright Patterson AFB, OH: Cockpit Integration Directorate, 1989.

27. Eimer, E.O., "Decision Aids: Disasters Waiting to Happen?" *in* Proceedings of IEEE 1987 National Aerospace and Electronics Conference, No. 3, pp 936-943, 1987b.

28. Lerch, C.E. and Prietula, M.J. "How Do We Trust Machine Advice?" *in* Proceedings of the 3rd International Conference on Computer-Interaction, pp 410-419, Amsterdam: Elsevier, 1989.

29. Riley, V. "Modelling the Dynamics of Pilot Interaction with an Electronic Crew, *in* Emerson, T.J., Reinecke, M., Reising, J.M. and Taylor, R.M. (Eds.) "The Human-Electronic Crew: Is The Team Maturing?" Proceedings of the 2nd GAF/RAF/USAF Workshop, PD-DR-P5, April 1991. RAF Institute of Aviation Medicine, Farnborough, Hants, April 1991.

30. Muir, B.M. "Trust Between Humans and Machines, and the Design of Decision Aids", Int. J. Man-Machine Studies, Vol. 27, pp 527-539, 1987.

31. Taylor, R.M. "Trust and Awareness in Human-Electronic Crew Teamwork. *in* Emerson, T.J., Reinecke, M., Reising, J.M. and Taylor, R.M. (Eds.) "The Human-Electronic Crew: Can They Work Together?" Proceedings of the 1st GAF/RAF/USAF Workshop, BSD-DR-G4, December 1988. RAF Institute of Aviation Medicine, Farnborough, Hants, December 1988. Tech Report WRDC-TR-89-7008 Wright Patterson AFB, OH: Cockpit Integration Directorate, 1989.

STATUS OF AUTOMATIC GUIDANCE SYSTEMS FOR ROTORCRAFT IN LOW ALTITUDE FLIGHT

Banavar Sridhar, Victor H.L. Cheng, and Harry N. Swenson
NASA Ames Research Center
Moffett Field, CA 94035

Abstract

Rotorcraft operating in high-threat environment fly close to the earth's surface to utilize surrounding terrain, vegetation, or man-made objects to minimize the risk of being detected by an enemy. The piloting of the rotorcraft is at best a very demanding task and the pilots need help from on-board automation tools in order to devote more time to mission-related activities. The Automated Nap-of-the-Earth (NOE) Flight Program is a cooperative NASA/Army program aimed at the development of technologies for enhancing piloted low-altitude/NOE flight path management and control through computer and sensor aiding. The long-term objective is to work towards achieving automation for aiding the pilot in NOE flight with a flight demonstration of resulting computer/sensor aiding concepts at an established course. The technology for pilot-centered NOE automation is not currently available. Success in automating NOE functions will depend on major breakthroughs in real-time flight path planning algorithms, effective methods for the pilot to interface to the automatic modes, understanding of visual images, sensor data processing/fusion, and sensor development. Our approach to developing the technologies required to solve this problem consists of the following phases: (a) algorithm development, (b) laboratory evaluation, (c) piloted ground simulation, and (d) evaluation in flight. This paper gives an overview of the research in this area at NASA Ames Research Center.

1 Introduction

The complexity of rotorcraft missions involving operations close to the ground in nap-of-the-earth (NOE) flight for long periods of time result in high pilot workload. This is especially true for single-pilot vehicles, such as was originally intended for RAH-66 Comanche. In order to allow a pilot time to perform mission-oriented tasks, some type of automatic system capable of performing guidance and control functions would be highly desirable. The problem of automating NOE flight, however, is extremely challenging due to the advances necessary in several technologies. This paper describes the status of research at NASA in three key areas: (a)Terrain Following/Terrain Avoidance, (b) obstacle detection, and (c) obstacle avoidance.

Guidance for NOE flight, as shown in Fig.1, can be divided into far-field, mid-field, and near-field requirements that can be envisioned as three dependent feedback loops [1]. The outer loop is responsible for far-field mission planning and waypoint selection, tasks that are generally performed prior to flight, but can benefit greatly from a real-time on-board re-planning capability. The middle loop governs terrain following and terrain avoidance about the nominal waypoint path. The outer and middle loop guidance, referred to as database-derived guidance, uses a priori information about terrain and threat leading to a trajectory planning algorithm. The database is unlikely to have information about objects such as trees, buildings, wires, etc., which may be in the flight path of the rotorcraft. This trajectory is modified by the inner-loop obstacle avoidance guidance based on information acquired by the on-board sensors and is a critical component of an automatic NOE system. Various on-going studies are addressing the issue of on-board sensors that could facilitate a pilot in detecting obstacles in the flight path. The obstacle detection problem is posed as the problem of finding ranges to all objects in the field of view based on measurements of the optical flow or stereo imagery. A complete obstacle detection system would require the integration of both active and passive sensors [2]. The Army has several programs to develop an active sensor, and we are emphasizing the use of passive electro-optical sensors. Such sensor systems could allow a pilot to concentrate less heavily on guiding the vehicle, thus allowing more time for attending weapons and communication systems.

The maturity of different technologies required to achieve the guidance tasks described above varies significantly. The Terrain Following/Terrain Avoidance (TF/TA) algorithms and display concepts used in database-derived guidance have reached a level where they can be tested in flight under operational conditions. The inner-loop obstacle avoidance algorithms have been developed in a graphics workstation environment and are ready for evaluation in piloted simulations. The vision-based obstacle detection technology is the least mature but we have succesfully applied passive ranging algorithms, for the first time, to images collected in a CH-47 helicopter flight.

The paper is organized as follows: Section 2 describes the guidance and display associated with the operationally ready TF/TA algorithms. Section 3 describes the inner loop obstacle avoidance guidance algorithm. Section 4 provides an overview of a maximally passive ranging system using elecrto-optical sensors. Finally, Section 5 provides summary, conclusions and ideas for future work.

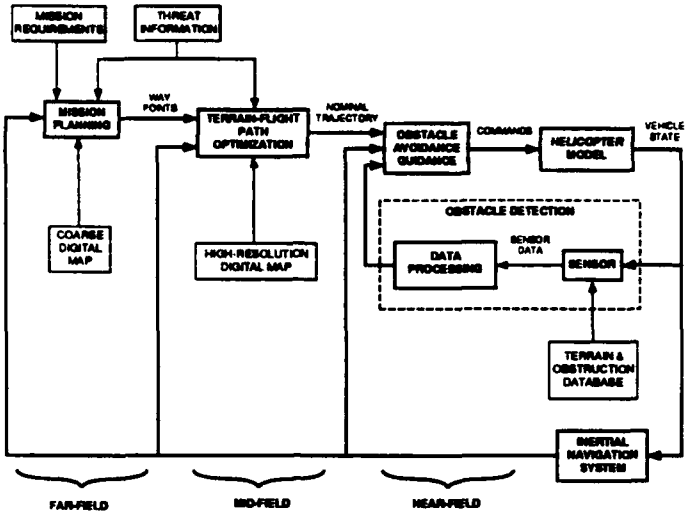


Figure 1: NOE Guidance Structure

2 TF/TA Guidance

Currently, rotorcraft operating in threat areas achieve low-level, maneuvering penetration capability during night-time and adverse weather conditions through the use of a combination of technologies such as terrain-following (TF) radar systems, forward looking infrared and night vision goggles. TF systems were initially developed for fixed-wing tactical and strategic aircraft [3] and provide vertical commands which can be displayed on a flight director for manual flight or fed to the flight control system for automatic flight. The extension of TF capability to include lateral maneuvering by taking advantage of on-board digital terrain data is commonly referred to as Terrain Following/Terrain Avoidance (TF/TA) in the literature. Within the last few years TF/TA algorithms have been modified to suit the requirements of rotorcraft [4]. Research at NASA has concentrated on incorporating these algorithms into an operationally acceptable system, referred to as the Computer Aiding for Low-Altitude Helicopter Flight (CALAHF) guidance system. Several piloted simulations [5] of the CALAHF guidance system have been conducted to develop the system and pilot interface and to evaluate pilot tracking performance and situational awareness under various flight and environmental conditions. The very favorable pilot feedback and response to these simulations [6] has led to a joint NASA and U.S. Army flight test of the CALAHF flight guidance system. The NASA-developed system will be flown on the U.S. Army Avionics Research and Development Activity's UH-60 STAR (System Testbed for Avionics Research) research helicopter.

2.1 System Description

Fig. 2 shows a functional block diagram of the CALAHF flight system. The three major components: (1) Dynapath trajectory-generation algorithm, (2) trajectory coupler, and (3) displayed information are discussed below.

Trajectory Generation Algorithm

Dynapath is a valley-seeking trajectory generating algorithm based on a forward-chaining dynamic-programming technique originally developed for the U.S. Air Force. Significant modifications have been made to this guidance algorithm to adapt it for manual rotorcraft operations. The algorithm uses two types of inputs. The first, characterized as mission dependent information, includes mission waypoints for defining a global trajectory to be flown, and Defense Mapping Agency digital terrain elevation data of the area in which the mission is to be accomplished. The second kind of input consists of pilot-comfort and aircraft-dependent parameters: maximum bank angle commands, maximum climb and dive angles, maximum pull up and push over load factor, set-clearance altitude (desired trajectory altitude above the ground) along with sensed aircraft state information.

Dynapath uses a decoupled procedure in which the lateral and vertical trajectory solutions are determined independently to obtain an optimal trajectory. In this decoupled procedure, the lateral ground track is first determined by assuming that the aircraft can maintain the vertical set-clearance altitude. The vertical trajectory is then calculated using aircraft normal load factor and flight path angle as maneuver constraints to maintain the aircraft at or slightly above the vertical set clearance as determined from the digital terrain map and the lateral ground track.

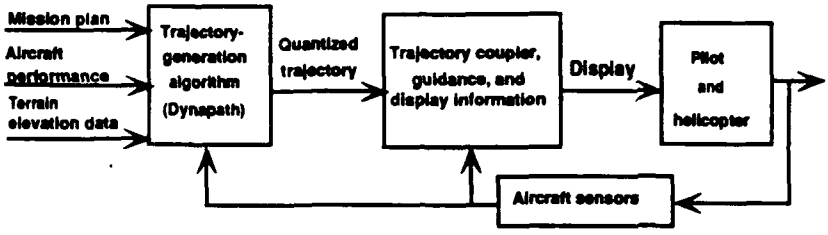


Figure 2: CALAHF system block diagram

The lateral path is calculated using a tree structure of possible two-dimensional trajectories by using discrete values of aircraft bank angle. Assuming constant speed and coordinated flight (zero sideslip), each discrete bank angle produces a possible path which in combination forms a tree of possible paths (Fig. 3). In this implementation, the bank angle control has five discrete values that are used for the trajectory calculation. The number of possible paths is reduced to a reasonable level by pruning. Pruning the tree after three to four levels of branching gave the best mix of branch generation and computational speed based upon results from non-real-time computer simulations. After the tree structure of possible paths has been propagated through the entire patch length, the cumulative cost (J) of all surviving branches are compared, and the path with the lowest cost is selected as the optimal trajectory.

The cost function J used to determine the optimal trajectory is

$$J = \sum_{i=1}^{30} H_i^2 + f(D)\omega D_i^2 + \alpha(\Delta\Psi_i)^2 \qquad (1)$$

where H_i is the altitude above sea level at node i, D_i is the lateral distance from reference path at node i, ω is the TF/TA ratio, $f(D)$ is a dead band on the lateral deviation cost, $\Delta\Psi_i$ is the error between reference and command heading at node i and α is the heading weight.

The main parameters in this performance measure are the terms representing altitude H and reference-path deviation D. The cost-functional, when driven by these two terms, allows lateral maneuvering to seek lower altitude terrain by the cost reduction from H; excessive deviation from the reference path is controlled by increasing cost due to D. The TF/TA ratio ω allows blending of these two terms to obtain a desired balance between vertical and horizontal maneuvering. The $f(D)$and $\alpha(\Delta\Psi_i)$ terms were added to reduce undesirable oscillations in the trajectory about the nominal path that are caused by the bank-angle quantization. The $f(D)$ eliminates the need for precise following of the reference path and the $\alpha(\Delta\Psi_i)$ term provides a penalty for changing the heading from that given by the reference path. These two terms were added as a result of experience gained in piloted simulations to make the trajectory-generation algorithm emulate pilot control strategies for low-altitude maneuvering flight.

The trajectory-generation algorithm, as defined above, is designed to compute guidance for a patch which is the area in front of the aircraft's present location. The patch width is the maximum lateral deviation, and the length is the flight preview distance. Both are input parameters selected by the user. The algorithm is computationally intensive: using representative values for patch length (30 sec) and maximum lateral deviation (1 km) the computational cycle is approximately 4 to 5 sec for a modern (1 to 2 MIP) flight computer. Although the algorithm is updated every cycle time, the updates are blended in such a way that a pilot sees a continuous path and the updates are imperceptible to him.

Trajectory Coupler

After the Dynapath algorithm produces its optimal trajectory it is passed to the trajectory coupler. The trajectory is represented by 30 discrete instances of commanded aircraft-inertial state (position, velocity and acceleration) at 1 sec intervals. Also stored are commanded bank angles, headings and vertical flight-path angles. The trajectory coupler, converts the quantized commanded trajectory into a trajectory command that is designed to work synchronously with the pilot displays at a minimum of 20 Hz, thus not imposing any time-delay that is perceptible to the pilot. This is accomplished by interpolating within the trajectory to determine the instantaneous position of the trajectory points to be presented on the pilots head-up-display. In the future, when the TF/TA guidance is integrated with the obstacle avoidance guidance (OAG) according to Fig. 1, the OAG will replace this trajectory coupler function.

Displayed Information

The guidance and control information is displayed to the pilot on a helmet mounted display (HMD) in the format shown in Fig. 4. The HMD format is a mixture of screen, body, and inertially referenced symbols. The screen referenced symbols include: a heading tape (021), engine torque (45%), airspeed (63 kts), radar altitude (105 ft), and ball and slip indicator. The body referenced symbols are the aircraft nose (><), and the flight-path vector/predictor. All remaining symbols are inertially refer-

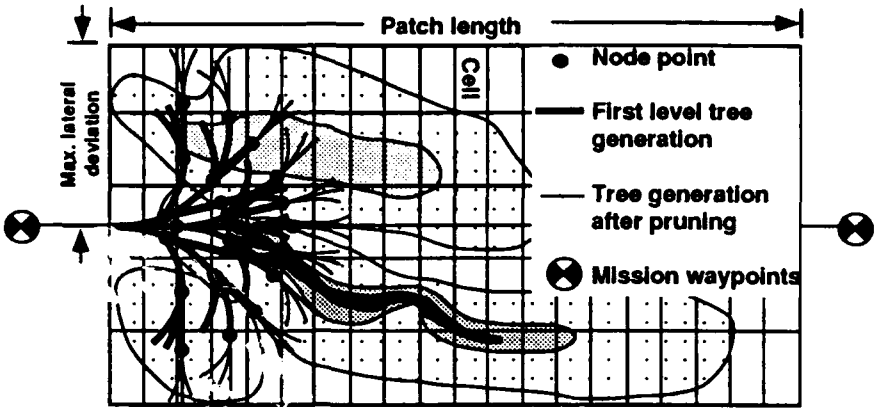


Figure 3: Dynapath tree generation

enced. The primary situational information is presented to the pilot with an inertially stabilized flight-path vector/predictor symbol predicting the rotorcraft location 4 seconds ahead, represented by the circular aircraft icon with attached airspeed flight director tape. The situational information presented on the HMD in Fig. 4 indicates the pilot is turning right with a slight descent as indicated by the flight-path vector/predictor below the horizon, and is looking approximately along the longitudinal axis of the aircraft as indicated by the position of the aircraft nose symbol.

The Dynapath trajectory information on the HMD is given by the pathway-in-the-sky and phantom aircraft. The pathway symbols represent a three-dimensional perspective of the inertial position and heading of the discretized Dynapath trajectory. The phantom aircraft is displayed on the HMD as a delta-wing aircraft. The phantom aircraft represents the instantaneous position along the Dynapath trajectory that is 4 seconds ahead of the pilot's aircraft. By positioning the flight-path vector symbol on the phantom aircraft, the pilot will track the desired trajectory. In Fig. 4, the HMD symbols are presenting a climbing right turn.

The pathway is 100 ft (roughly two rotor diameters) wide at the bottom and parallel to the horizon with vertical projections that are canted at a 45 angle; the width at the top is 200 ft. The depth of the path is 50 ft below the intended trajectory; thus when flying a level straight-line commanded path, the pilots used the analogy of traveling in a full irrigation canal for describing the pathway symbols. Fig. 4 shows the baseline configuration of 7 lines.

2.2 Piloted Simulation

There have been five piloted simulations dedicated to the development and evaluation of the computer aiding for low-altitude helicopter flight guidance concepts [5, 6]. The simulations have been conducted on the Ames Research Center six-degree-of-freedom Vertical Motion Simulator (VMS). The VMS provides extensive cockpit motion for use in evaluating handling qualities associated with advanced guidance concepts for existing and proposed aircraft. Based on the system performance and pilot acceptance demonstrated during the third simulation, the concept was believed to be ready for flight evaluation, both as a first step in initiating Automated NOE flight research and a standalone capability to meet the operational military requirements for covert low-altitude penetration. This resulted in an agreement between NASA-Ames and the U.S. Army Avionics Research and Development Activity (AVRADA) for a joint flight experiment in the AVRADA UH-60 STAR helicopter. The final two simulations were conducted in direct support of the flight test program. In the fifth simulation, 5 NASA and Army project pilots flew over 300 simulation data runs evaluating and defining the system throughout the proposed flight test envelope. Eighteen guest and evaluation pilots from NASA, DOD and U.S. industry flew the system, giving highly favorable feedback on the system development. The evaluation pilots were able to manually track the HMD guidance through various combinations of terrain, speeds, and weather representative of system use. The guidance can be followed with low pilot workload without detracting from his awareness of the outside world. The pilot was able to combine the guidance with his visual senses to optimize the mission success in varying weather/threat conditions.

3 Obstacle Avoidance Guidance

The obstacle avoidance guidance (OAG) assumes that the database-driven guidance would provide the nominal course to be followed. The OAG is motivated by the recognition that it is unreasonable to expect the database to be precise and up-to-date enough to contain detailed obstacle information. Neither would it expect the obstacle-detection system to be powerful enough to provide the locations and sizes of all the obstacles. Specifically, the sensors would not be able to detect obstacles hidden behind other objects, and the obstacle-avoidance guidance system has to operate within such

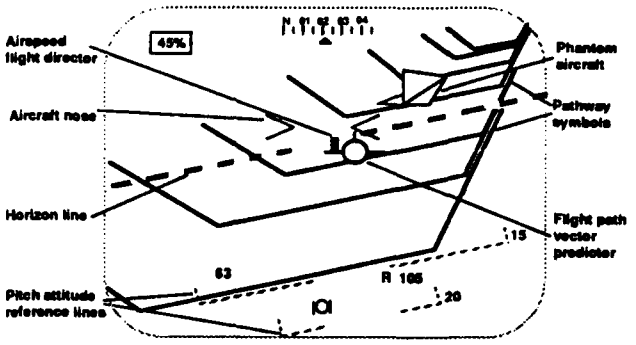


Figure 4: Helmet Mounted display format

a limitation. We assume that the sensors can provide a sparse range map within the sensors' field of regard. Our approach involves selective augmentation of the sparse range data with measurements from active ranging sensors to assure safe flight [2]. The guidance does not use pre-designed discrete obstacle avoidance maneuvers but instead makes continuous adaptations to the flight path based upon updated sensor range information. Further, the guidance does not assume a-priori knowledge of the distinction between obstacles and terrain, but instead accomplishes this differentiation using real-time computation as would be necessary in actual flight.

3.1 Input Data Preprocessing

Course-following guidance is accomplished by tracking an imaginary reference point that is placed ahead of the vehicle on the nominal path by a distance proportional to the rotorcraft's velocity. Fig. 5 contains a block diagram of the OAG structure. During each guidance cycle, the obstacle detection system provides three-dimensional range information in body-fixed coordinates. This 3-D range map simply provides the numerical distance from the rotorcraft to terrain or obstacles, currently modeled in the simulation to be available within a 60° field of view in both azimuth and elevation. To emulate passive sensor characteristics, range information is stored by perspective projection in a 128 × 128 pixel array representing the focal plane of a FLIR or TV image. From the body-fixed reference frame, the 3-D range information is transformed to inertial coordinates and stored. At any time, the derived inertial database includes terrain and obstacle data within 350m longitudinally and laterally to the vehicle position. Based on the inertial database, a simple test is performed to differentiate obstacles from terrain. This test involves examining the 3-D data base, which is altitude as a function of horizontal displacement, for large changes in altitude. Upon differentiating the terrain and obstacle data, the obstacle data is used to construct a 2-D range map while the terrain data is stored to provide terrain following guidance.

3.2 Flight Path Selection

Path selection begins by determining whether any obstructions lie in a direct path to the reference point, where the path is at least as wide as the rotor diameter with additional predefined clearance. If a clear path is available, a velocity command is given in the direction of the reference point. If obstructions exist, the 2-D inertial range map is inspected for an alternate route. A potential path from the range map is indicated by a discontinuity in range vs. azimuth angle [7]. A further requirement for a potential route is that the difference in range on either side of the discontinuity must be greater than the rotor diameter in order to allow for turning into the opening (see Fig. 6). If more than one possible opening is found, the selection is based on the path that minimizes the deviation from the nominal course. Once an opening is found, a velocity command is given along the open direction. In some instances there may be no open path available, in which case the vehicle is commanded to initiate a pedal turn towards the reference point. Upon facing the reference point, a velocity command to the reference point is given along with climb commands that allow the rotorcraft to climb with constant heading to clear the terrain and obstacles, i.e. the vehicle switches to contour flight. The terrain data from the inertial database along the commanded 2-D path provides the necessary information to generate commands in the vertical plane.

These algorithms have recently been implemented in a workstation-based simulation for evaluation. Fig. 7 contains an example of an automatically guided flight path about the nominal trajectory in white. Details of the guidance and control implementation will be reported in a future paper.

4 Vision Based Obstacle Detection

The vision system is intended to provide the on-board information necessary to modify the nominal trajectory of the rotorcraft provided by far-field and mid-field planning. NASA research focuses on identifying and developing the key components of a vision based obstacle detection capability for rotorcraft NOE flight. The components of an obstacle detection system is shown in Fig. 8. We assume that one or more passive electro-optical sensors is installed on the rotorcraft. In addition, we assume that an inertial navigation system and other sensors will be available to provide the rotorcraft velocity and body rates. Because vision alone will not be adequate for de-

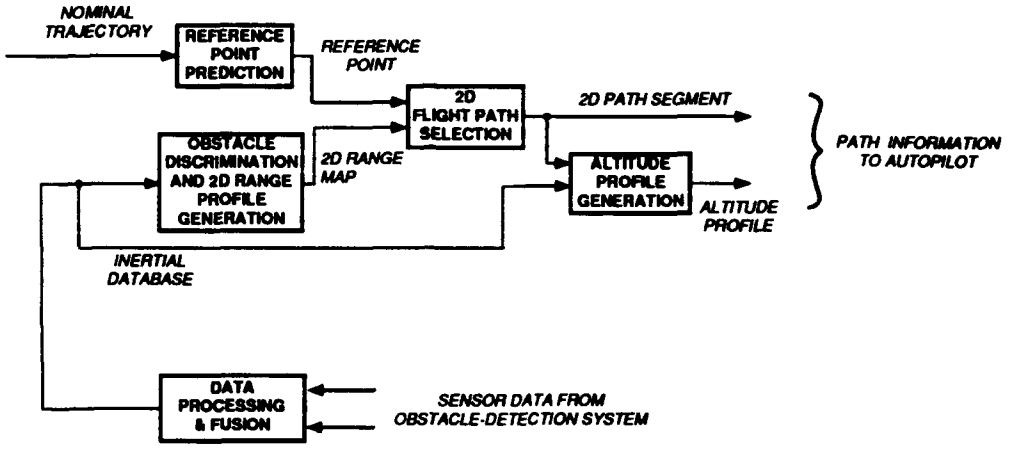


Figure 5: Obstacle avoidance guidance structure

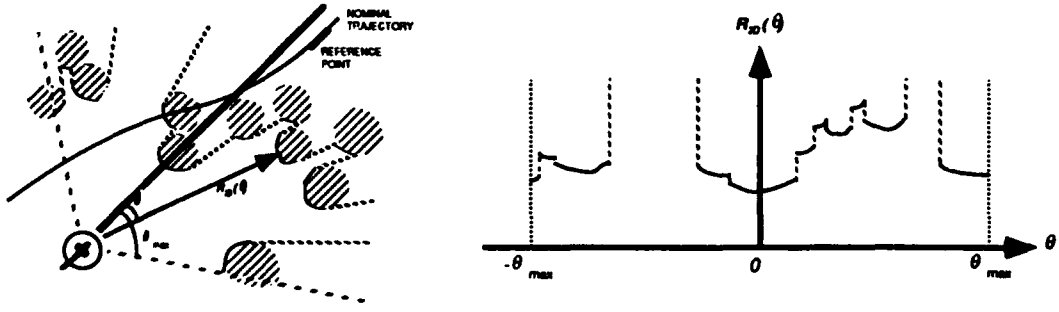


Figure 6: Two-dimensional obstacle avoidance

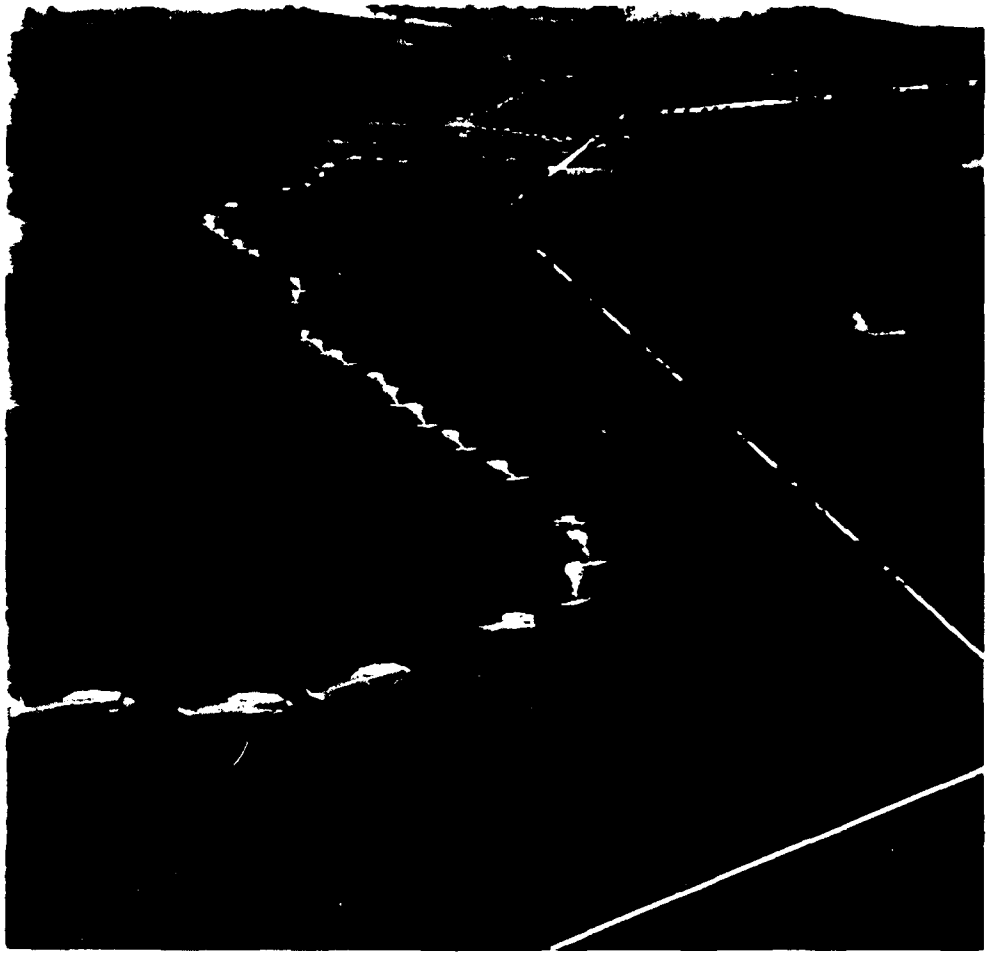


Figure 7: Example of automatically guided flight path

tecting small obstacles such as wires, it is expected that the system will include an active ranging sensor whose search area can be directed to complement the vision system to minimize detectability [2]. Rotorcraft maneuvers, natural scenery, and the availability of accurate vehicle position and attitude measurements motivate a new look at vision algorithms developed for robotics and machine inspection.

The obstacle detection system requires finding ranges to all objects in the field of view using a sequence of images. This problem can be broken into two steps. The first step consists of the computation of optical flow. The second step involves the use of the optical flow to estimate range. The computation of optical flow requires the determination of the displacement of image points over a sequence of images. The main difficulty in the computation is due to the assumption that an object in the terrain space corresponds to a unique point in the image. In an actual image, an object on the ground is more likely to be a region in the image. Another complication in the computation of the optical flow, referred to as the correspondence problem, results from the ambiguity in identifying features in two images that are projections of the same entity in the 3-dimensional world.

The computation of optical flow using two successive images has been an active area of research in computer vision. The methods used can be divided into two categories:

1. The feature-based approach computes ranges to explicitly identified features in the images, which can be points, lines, contours or any other clearly distinguishable part of the image. The extraction of features [8] in an image involves several operations such as enhancement, edge detection, linking and scene analysis. A fundamental assumption of this approach is that correspondence between features in two images needs to be established prior to computation of range.

2. The field-based techniques assume a continuous

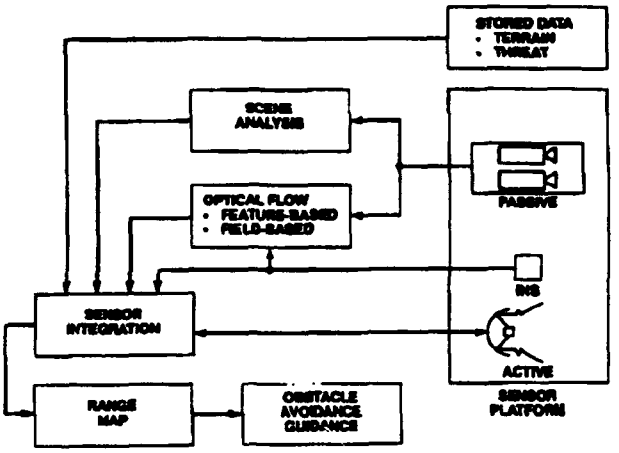


Figure 8: Components of an Obstacle Detection System

variation of image intensity as a function of position and time. This approach was introduced by Horn [9] and provides a dense map of the optical flow. However, the computational experience with this approach is limited to simple scenes and the accuracy of available algorithms based on this method is very susceptible to noise.

Our feature-based algorithms are based on feature tracking followed by recursive range estimation using an extended Kalman filter [10, 11, 12]. They differ from earlier approaches in several different ways. The extraction of range information and the simultaneous estimation of the observer (rotorcraft) motion parameters (translational and rotational velocity), using only image data, leads to ill-conditioned computations. In the case of the rotorcraft, the motion parameters can be measured independently using an on-board navigation system; consequently the use of image data only to extract range information results in more stable computations. The knowledge about the rotorcraft motion is used to reduce the amount of computation in establishing the correspondence between features as well as to discard false matches. All the algorithms are structured such that they can handle a sequence of images in a recursive manner and, unlike several algorithms reported in the literature, are not limited to linear motion of the rotorcraft.

We are developing a field-based algorithm based on the direct computation of range. The equation for range is derived by combining local Taylor series expansion for image irradiance in a pair of images with the geometry of perspective projection. This ranging equation relates the image irradiance gradients and the camera and motion parameters to range. This process also yields a second equation that predicts the error involved in the approximation. An optimal method for satisfying the ranging equation is then developed for computing range [13]. In addition, we are considering (a) the application of 3-dimensional Fourier transform to a sequence of images resulting in a shift and add algorithm [14] and (b) an intensity gradient approach [15]. The field-based methods are developed for linear helicopter motion and are being modified to handle general helicopter motion.

We are also exploring the use of two or more imaging sensors (stereo) to provide better range information near the focus of expansion [18]. The Kalman filter formulation allows the use of several electro-optical sensors and it provides for a natural way of integrating stereo and motion methods. In addition, since quantitative vision based algorithms do not provide range to all points in the field of view (FOV), we are also investigating the use of qualitative vision algorithms such as scene analysis to fill the gaps in the range map [19].

A major issue in the implementation of vision algorithms is the trade off between computational speed and the denseness of the range map. Due to the changing nature of the hardware and our desire for developing portable software, until recently, we have addressed this problem only at the algorithmic level. We plan to achieve the goal of real-time implementation by using parallel computation. Currently, the feature-based passive ranging algorithm is being restructured to exploit the inherent parallelism in the algorithm and for implementation in a iWARP parallel computer system [20]. These issues will be studied in-depth with the end result of developing a real-time vision system.

4.1 Evaluation of Range Algorithms

We plan to evaluate all our algorithms using image sequences acquired in the laboratory and in flight. An experiment has been designed to acquire a sequence of images in the laboratory by using a camera mounted on a 3 degree-of-freedom motion table. The camera position and orientation is controlled by a computer to achieve desired curvilinear camera motion. The camera can be moved at different speeds. This experiment can be scaled to simulate a helicopter flying at 20 knots where the ob-

jects in the FOV vary from 50m to 500m and the images are processed for range computation every 0.25 seconds. Fig. 9 shows the image processing laboratory setup. A detailed description of the hardware and software used in the acquisition of laboratory images, evaluation of vision algorithms for range estimation, and visualization of the results can be found in [16]. Results of processing the laboratory images indicate that it is posssible to estimate range to an accuracy of 5 to 10% [12].

NASA Ames has recently developed a database from CH-47 helicopter flight data including imagery, rotorcraft attitude and position time histories, camera parameters, and ground truth range measurements [17]. The flight experiment designed to collect the necessary data is depicted in Fig. 10. The video camera was rigidly mounted under the rotorcraft nose and oriented roughly along the direction of flight so as to observe designated obstacles of interest which the rotorcraft would encounter. True range measurements were obtained using a ground-based laser tracker. We have checked the flight data for consistency using a state estimation technique. The calibration of the camera in an operational system required modification of camera calibration algorithms. The database includes typical flight profiles such as straight and level flight, banked and curved flight, pedal turns, bob-ups, and transitions to and from hover. The flight database will be available to vision researchers to evaluate algorithms.

Fig. 11 shows the first, 25th, 50th, and the last image in a sequence of 75 images from the database. These images were collected at a frame rate of 30/sec (33 milliseconds between successive images). The helicopter is flying straight above the runway at an altitude varying between 13 and 14 feet and speed varying between 18 to 20 knots. The helicopter covers a distance of about a foot between images. The laser tracker was used to measure the distances to the trucks A, B, C, and D parked along the runway. This information is used to generate the truth data shown in Table 1 relative to the helicopter location at the 75th image. The true range varies by ± 10 feet depending on different parts of the truck.

Features in the images were tracked and their positions provided to the extended Kalman filter. There are several features associated with a single object on the ground. We computed the mean and standard deviation of the range estimates for all features corresponding to a single object. These results are shown in Table 1. The range estimates agree well with the true range for trucks A, C, and D. Note that a portion of the standard deviation in the estimated range is due to the fact that different parts of a truck are at different ranges from the camera. The range estimate to the truck B is less accurate, having an error of 13.3% and a standard deviation of 72 feet, both larger than the quantities for the other trucks. The reasons for the poor performance of the range estimate algorithm for truck B are: (a) truck B is farther from the sensor than trucks A, C, and D; (b) truck B is only about 5 degrees away from the FOE (which is shown as an "X" in the lower right image at Fig. 12). The estimation error for an object like truck B can be reduced by the use of stereo.

In summary, we have presented for the first time results on the feasibility of using vision-based algorithms for acquiring range information using images acquired from helicopter flight. Our contributions are twofold: (a) development of a flight and image database for verification of vision based algorithms and (b) a passive ranging methodology tailored to the needs of helicopter flight. Our preliminary results indicate that it is possible to get adequate range estimates except at regions close to the FOE. As we get closer to the FOE, the error in range increases since the magnitude of the disparity gets smaller resulting in a low signal to noise ratio. We plan to continue the evaluation of the method using several different flight scenarios namely (a) helicopter approaching the runway while performing a turn and bank maneuver, (b) helicopter performing a bob-up, (c) other scenarios for which we have truth data, and (d) scenarios with different contrasts between the objects and the background.

Table 1: Passive Range Estimates using Flight Images

Truck	True Range ft	Est Range ft	Std Dev ft	%error
A	325	324	25	0.3
B	576	499	72	13.3
C	451	452	50	0.2
D	205	215	10	4.8

The performance of the motion-based passive ranging method can be improved near the FOE by using stereo (two or more cameras). We have developed an integrated stereo and motion algorithm using the EKF. The use of stereo has additional benefits. Stereo provides range information when the helicopter is stationary and, further, the stereo range can be used to initialize the EKF. Passive ranging algorithms, in general, provide range information only in regions of the image where there is a variation in image intensity. The resulting sparse range map can be improved using other vision based methods such as texture analysis and segmentation to fill gaps in the range map based on optical flow.

5 Concluding Remarks

Research is continuing at NASA Ames in the development of automation tools for rotorcraft low-altitude flight. We have focused our attention on (1) TF/TA guidance, (2) obstacle avoidance, and (3) vision-based obstacle detection. Our intent is to test each of the above technologies in both laboratory and in flight. Of the topics discussed above, the database-derived guidance is most mature and a flight test is being planned for the near future to evaluate the pilots ability to manually track the TF/TA guidance and its impact on pilot workload and situational awareness. The obstacle avoidance research has led to the development of a three-dimensional obstacle avoidance guidance methodology which makes realistic assumptions on the capabilities of active and passive sensors. The sensor-derived obstacle

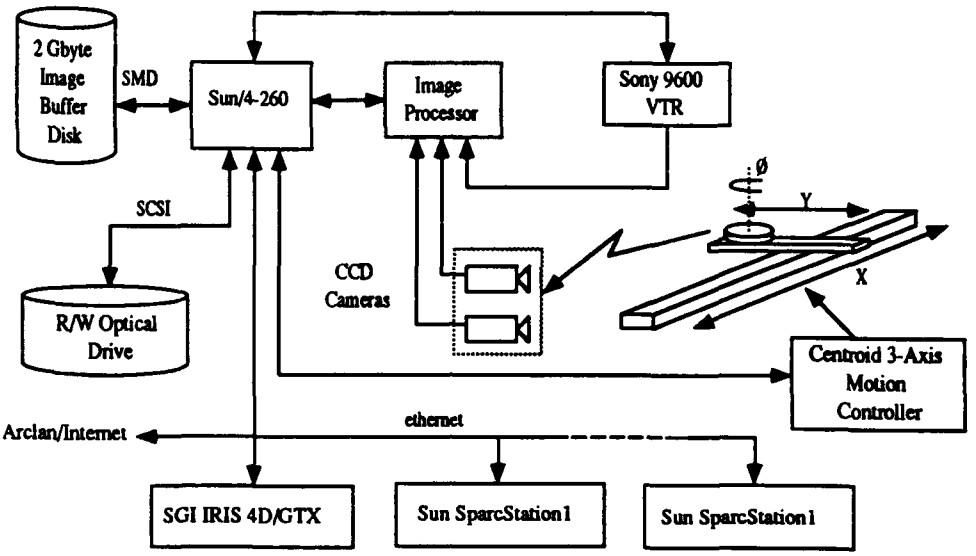


Figure 9: Image Processing Laboratory Setup

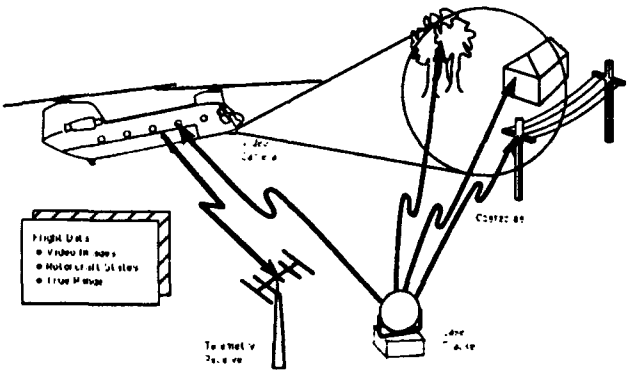


Figure 10: Flight Experiment Setup

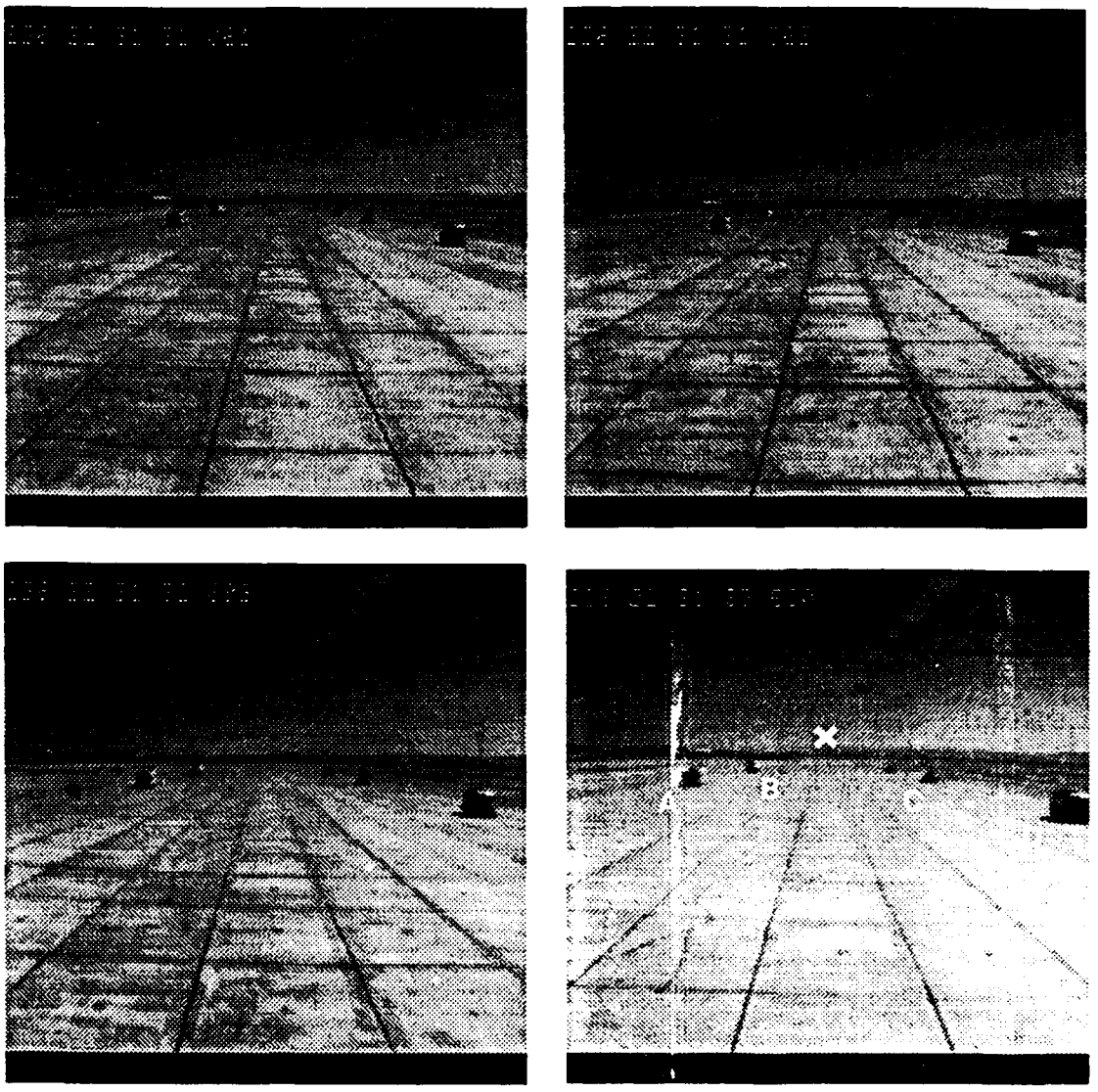


Figure 11. Sample Image sequence from Flight Test

avoidance guidance, with an active sensor in the initial phase, will need further refinement to make it more adaptive to the unpredictable environment and acceptable to pilots before a flight test can be contemplated. The vision-based obstacle detection system approach has resulted in a range estimation technique which can handle general helicopter manuevers and large variations in the optical flow encountered during flight. We have demonstrated the first application of vision-based methods to helicopter flight data. The obstacle detection system based on eletro-optical sensors is an extremely complex problem and requires further evaluation using additional flight data and substantial effort towards achieving real-time implementation. We are also considering near-term applications of the above technologies for simplified situations.

References

[1] Cheng, V.H.L., and Sridhar, B., "Considerations for Automated Nap-of-the-Earth Flight," *Proceedings of the1988 American Control Conference*, Atlanta, GA, June 15-17, 1988.

[2] Cheng, V.H.L., and Sridhar, B., "Integration of Active and Passive Sensors for Obstacle Avoidance," *IEEE Control Systems Magazine*, Vol. 10, No. 4, pp. 43-50, June 1990.

[3] Russ, D.E., Houtz, J.E., and Rothstein, S.W., "Evaluation of Alternative Terrain Following / Terrain Avoidance (TF/TA) Systems Design Using Pilot in the Loop Simulation," *AGARD CP-387*, Oct. 1985.

[4] Pekelsma, N.J., and Denton, R.V., "Pilot Oriented Aids for Helicopter Automatic Nap of the Earth Flight," *AHS Proceedings National Specialists' Meeting in Rotorcraft Flight Controls and Avionics*, Oct. 1987.

[5] Swenson, H.N., Hardy, G.H., and Morris, P.M., "Simulation Evaluation of Helicopter Terrain Following/Terrain Avoidance Concepts," *8th Digital Avionics Systems Conference*, Oct 17-20, San Jose, CA, 1988.

[6] Swenson, H.N., Zelenka,R.E., Hardy, G.H., and Dearing, M.G.,"Simulation Evaluation of a Low Altitude Helicopter Flight Guidance System Adapted for Helmet-Mounted Display," *10th Digital Avionics Systems Conference*, Los Angeles, Oct 14-17, 1991.

[7] Cheng, V.H.L., "Concept Development of Automatic Guidance for Rotorcraft Obstacle Avoidance," *IEEE Trans. Robotics and Auto.*, Vol. 6, No. 2, pp. 252-257, April 1990.

[8] Ballard, D.H., and Brown, C.M., *Computer Vision*, Prentice-Hall, Englewood Cliffs, N.J., 1982.

[9] Horn, B.K.P., *Robot Vision*, The M.I.T Press, Cambridge, MA, 1986.

[10] Sridhar, B., Cheng, V.H.L., and Phatak, A.V., "Kalman Filter Based Range Estimation for Autonomous Navigation using Imaging Sensors," *Proceedings of the 11th IFAC Symposium on Automatic Control in Aerospace*, Tsukuba, Japan, July 89.

[11] Sridhar, B., and Phatak, A.V., "Simulation and Analysis of Image-based Navigation System for Rotorcraft low altitude flight," *Proceedings of AHS National Specialists' Meeting Automation Applications of Rotorcraft*, Atlanta, GA, April 4-6, 1988. To appear in *IEEE Trans. Systems, Man and Cybernetics*

[12] Sridhar, B., Suorsa, R., and Hussien, B., "Passive Range Estimation for Rotorcraft Low Altitude Flight," *NASA TM*, October 1990. To appear in *Machine Vision and Applications*

[13] Menon, P.K., and Sridhar, B., "Image Based Range Determination," *Proceedings of the AIAA Guidance and Control Conference*, Portland, OR, August 1990.

[14] Barniv, Y., "Application of Velocity Filtering to Optical Flow Calculations," NASA TM, April 1989.

[15] Skifstad, K., and Jain, R., "Range Estimation from Intensity Gradient Analysis," *Machine Vision and Applications*, Vol. 2, pp. 81-102, 1989.

[16] Suorsa, R., and Sridhar, B., "Validation of Vision-Based Obstacle Detection Algorithms for Low Altitude Flight," to be presented at the *SPIE International Symposium on Advances in Intelligent Systems*, Boston, MA, Nov 1990.

[17] Smith, P., "Flight Data Acquisition for Validation of Passive Ranging Algorithms for Obstacle Avoidance," *Proceedings of the American Helicopter Society Forum*, Washington, D.C., May 1990.

[18] Sridhar, B., and Suorsa, R., "Integration of Motion and Stereo Sensors in Passive Ranging Systems," *Proceedings of the 1990 American Control Conference*, San Diego, CA, May 1990.

[19] Sridhar, B., Phatak, A.V. , and Chatterji, G., "Passive Range Sensor Refinement Using Texture and Segmentation," *AIAA Conference on Aerospace Sensing*, Orlando, FL, Apr 1991.

[20] Borkar, S., "iWARP: An Integrated Solution to High-Speed Parallel Computing," *Proc. Conference on Supercomputing 88*, Orlando, FL, Nov 1988.

Knowledge-based planning for controlled airspace flight operation as part of a Cockpit Assistant

T. Prevot, R. Onken

Universität der Bundeswehr München
Institut für Systemdynamik und Flugmechanik
Werner Heisenberg Weg 39
D-8014 Neubiberg

H.-L. Dudek

Dornier Luftfahrt GmbH
Postfach 1303
D-7990 Friedrichshafen

Summary

A knowledge based computer aid for flight planning tasks as part of a Cockpit Assistant for IFR (Instrument Flight Rules) operation is presented in detail after a brief overview of the modular structure of the Cockpit Assistant has been given.

Based upon the requirements of air traffic management the system is aimed at supporting the pilot in complex planning and decision situation. By assessing the situation flight goal conflicts are detected and avoided by offering sensible flight plan modifications to the pilot. The main features of the situation assessment and planning module are discussed with regard to knowledge representation and applied solution model. Thus, conflict detection, selection of an alternate destination and rerouting algorithms are described in detail.

The present state of implementation, as tested in a flight simulation facility at the University of the German Armed Forces in Munich is presented as well as some results of the test phase with professional pilots.
The possibilities of integrating the system into the avionics of modern aircraft are discussed and an outlook is given of expanding it towards 4-D and RNAV capabilitiy and/or optimal 3-D aircraft trajectory planning under constraints from terrain, traffic and weather.

1. Introduction

Modern civil aviation essentially consists of flights under Instrument Flight Rules (IFR) to keep it indepedent of meteorological conditions. Highly developed aircrafts and the complexity of Air Traffic Management (ATM) require the pilot's attention especially in critical situations. System failures, weather changes as well as changing constraints from ATC may overstress the pilots, in the wake of which wrong decisions and accidents possibly occur.

Research on aircraft accidents [1] and the knowledge of human information processing, like findings of Rasmussen [2] among others, has shown that one major aspect of cockpit assistance should be the support in situation assessment and replanning in flight. The well-structured IFR problem space offers best premises for computer-aided problem detection. Knowledge of the flight plan and the overall situation enables the system to detect conflicts and modify the flight plan sensibly using the

principle of problem reduction. The necessity of basic plan modifications can be carried out in the same manner, whereas decisions, subject to uncertain and fuzzy criteria, require additional algorithms.

For complete situation assessment and evaluation monitoring of the plan execution as well as of the aircraft systems and environment is required. This includes the ability of the system to communicate with the pilot and ATC. Therefore the need for other modules and interfaces is given, resulting in a modular structured Cockpit Assistant, which comprises these features. This Assistant for Single-Pilot IFR Operation (ASPIO) has been implemented and tested in a flight simulator with good acceptance by the pilots [3] [4]. Figure 1 shows the integration of the Cockpit Assistant into the air traffic system.

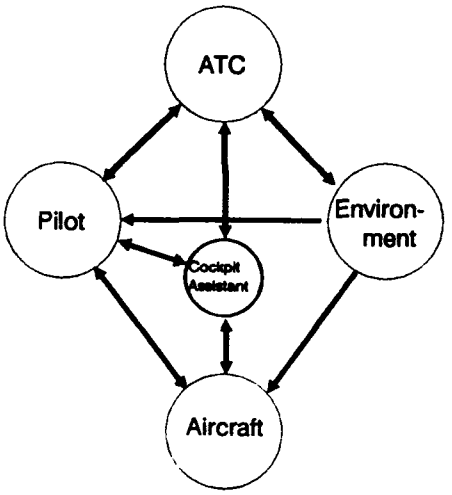


Figure 1 : Integration of the Cockpit Assistant into the air traffic system

2. Structure of the Cockpit Assistant

2.1. Modular structure and information flow

In figure 2 the major modules and the information flow of the Cockpit Assistant are presented. Additional modules offering special services and executional aids to the pilot are left out of this representation, although they proved to be quite helpful to the pilot. The complete modular structure of ASPIO can be taken from figure 9.

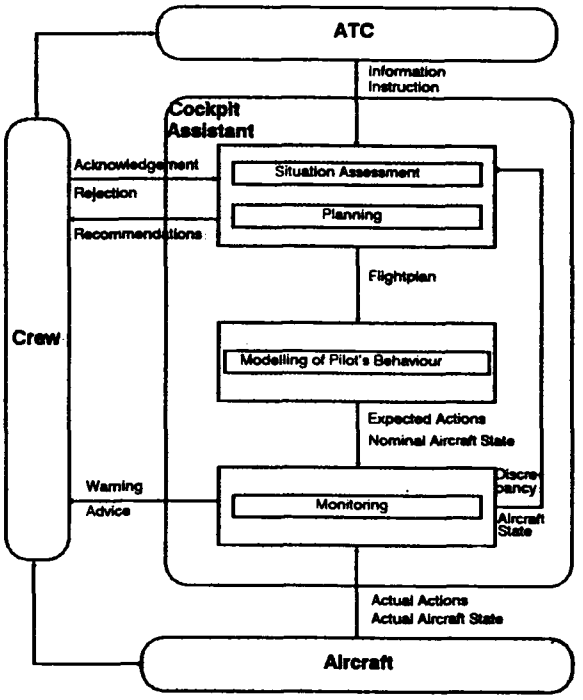


Figure 2 : Structure and information flow of the Cockpit Assistant

During a flight, where no problems occur and no replanning is necessary, the pilot will not become aware of the CA modules activity. The need for a flight plan modification results from the situation evaluation. When a new plan is generated, it is recommended to the pilot. If the pilot rejects the plan, the next best alternative will be presented. This procedure continues until the pilot acknowledges a plan. The new flightplan is the input for the following module, which tries to model the pilot's rule-based behaviour: The expected sequences of actions that are necessary to follow the flightplan.

These expected actions establish the input for the monitoring module. This module compares the nominal aircraft state, according to flightplan and expected pilot actions, to the actual aircraft state. When a discrepancy is detected, a hint is given to the pilot, so that small errors can be corrected immediatly and fatal errors can be avoided before they occur.

2.2. Interfaces to pilot and ATC

The pilot interface makes use of the most natural communication medium between humans: Speech. Speech output is used especially for recommendations, warnings and advice. In addition to speech output basic flightplan modifications are presented on a visual display. Speech input is used for acknowledgement and rejection and as one communication medium for instructions to the executional aids of the Cockpit Assistant. For speech input a speaker-dependent speech recognition system is used based on the phraseology of civil aviation.

For ATC communication a digital data connection has been assumed. During the experimental phase ATC commands have been entered into the computer system manually.

3. Situation assessment and flight planning module

3.1. Structure

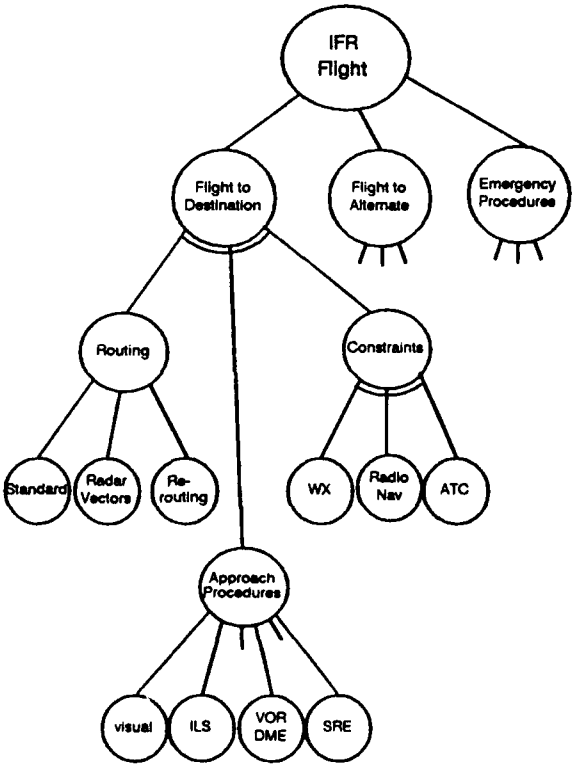


Figure 3 : AND/OR graph of an IFR-approach

The basic knowledge about task sequences during an IFR flight is represented as AND/OR graph, which is shown in figure 3 for the approach part of the flight.

The prevailing goal is to perform an IFR flight with the first priority to fly from a starting location to a predetermined destination. If this cannot be achieved, second and third priorities are to fly to an alternate destination or safely performing emergency procedures. For each of these possibilities several subgoals exist. The main aspects for the flight to any destination can be summarized as follows:

- a route from the actual position to the destination has to be selected, which can be
  - a standard route taken from the navigational data base
  - a radar vectored route guided by ATC instructions, where the point of return into the flightplan is uncertain
  - a self generated route planned by special rerouting algorithms
- at least one instrument approach procedure has to be practicable with respect to system status and weather conditions
- additionally the following constraints have to be taken into account for every leg of the flight:
  - appropriate weather conditions
  - possibility of navigation on the basis of the available radio nav facilities
  - ATC instructions

For the case that there is no solution for the node "flight to destination" first an alternate has to be selected before it can be treated in the same manner as the destination. So the

selection of an alternate destination is another major aspect of replanning the flight.

Thus, the ASPIO module for situation assessment and planning can be divided into several homogeneous submodules presented in figure 4 and explained in the following paragraphs.

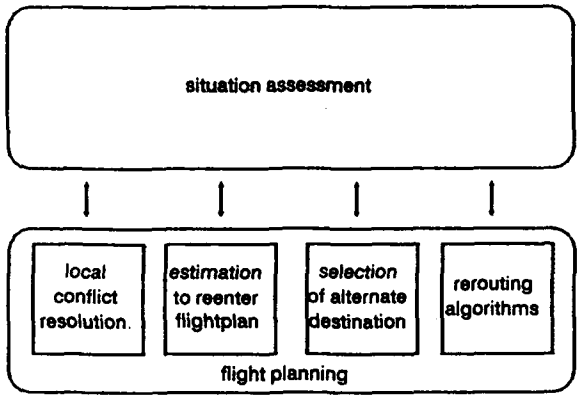


Figure 4 : Structure of the situation assessment and flight planning module

3.2. Situation assessment as application of problem reduction

As mentioned before, problem reduction is used for situation assessment. An excerpt of the knowledge base used for situation assessment has been given in figure 3. The technique consists of dividing an initial problem into a set of subproblems or subtasks. The knowledge representation as AND/OR graph is easing the application of problem reduction.

In [5] it is shown, that each node of an AND/OR graph is solved if

- it is an OR node and one of its successors is solved
- it is an AND node and all of its successors are solved
- it is a primitive, which is solved.

If no solution for a node can be found, the next higher OR node, has to be solved. Thus, alternatives can be acquired by processing a situation AND/OR graph. The provision of alternatives enables local conflict resolution after the conflict has been detected. Therefore the searching process in the AND/OR tree is the superior algorithm for conflict detection and coordination of planning, e.g. classifying the detailed needs for planning.

The searching strategy should be chosen with respect to computing time and effort, so that often a heuristical approach is recommended [6][7]. In order to apply a certain strategy a sequence for processing the successors of a node has to be defined. The selection of the "next best alternative" as the next OR successor to be processed is easy, if a situation independent ranking can be found. However, most basic flightplan modifications, including alternate and route selection, are strongly dependent on the actual situation. In this case the evaluation of alternatives has to consider the pilot's wishes as well as the actual system and environmental state, which might be fuzzy or uncertain.

3.3. Basic flightplan modifications

3.3.1. Selection of an alternate destination

If the node "flight to destination" cannot be solved, the best alternate has to be selected. Since the flight planning aid is to support the pilot, a ranking list of practicable alternate destinations is generated. The pilot has the final choice after the system has evaluated each of them and has presented them to the pilot in ranking order.

There are a number of criteria to be considered for the evaluation of the alternate destination, concerning

- runway length
- weather conditions with respect to wind, ceiling and visibility
- radio aids

- distance to actual position and destination.

- servicing capabilities

Since all of these criteria are imprecise or fuzzy, the evaluation process is based on fuzzy set theory first introduced by [8]. For each criterion a membership function can be defined. It shows the degree of membership for a fuzzy set of elements, which have the attribute of the predefined criterion in common. The degree of membership is defined from 0 to 1.0. The value of 1.0 is identical to the highest score of membership, e.g. the best criterion fit. Figure 5 shows a possible membership function for the criterion "long runway".

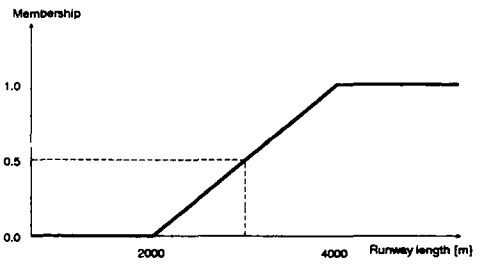


Figure 5 : Example for membership function

An airport with a runway length of 3000 m would have the membership degree of 0.5 within the fuzzy set of airports with "long runways" in this example. According to former explanations this membership function is dependent on the actual situation, because the minimum runway length varies due to estimated landing weight, aircraft condition and weather conditions at the airport.

Since there are more than one evaluation criteria, their degrees of memberships have to be composed to get a total ranking value for each alternative. Considering that not all criteria are equally important, their degree of membership is weighted. Sufficently good results have been gained by just adding up the weighted values of degrees of membership in order to compute the total ranking value. However, there are other possibilities [9][10], too.

3.3.2. Rerouting algorithms

For the case that an alternate has been selected or the planned route conflicts with disturbing events like areas of bad weather, the route has to be modified. To offer a practicable new route, all possible major conflicts, which cannot be solved locally, have to be taken into account by generating the route. Thus, the rerouting procedures are also based on a detailed situation analysis and evaluation of each flight-leg of the route.

A flight-leg is defined as the connection from one waypoint to the next. So, the principle is to connect given waypoints and to evaluate the resulting leg. The evaluation process is similar to the explained alternate evaluation. The main aspects for route planning are

- to keep away from bad weather area

- not to lose too much time

- not to consume too much fuel, at least to reach the destination

- to follow ATC-instructions

The first aspects are considered by evaluating criteria like distance to areas of bad weather, course deviation for each leg and total distance. The replanning area is kept small in order to avoid conflicts with ATC constraints.

It is important for an IFR-flight to assure 'radio nav' capability for the flight. So, the possible waypoints are restricted to radio aids and the legs can only be as long as the beacons can be kept in range. The applied rerouting algorithms offer conventional 'radio nav' as well as 'area nav' capability.

Only a problem for 'area nav' might appear, if the navigational data base does not offer appropriate waypoints for the area to plan and the planned route includes an unsolvable conflict. For this case, waypoints must be generated. This is done by computing legs that are tangent to an estimated safety range around the center of the conflict area. Figure 6 shows the planning on waypoints from the data base and generating new waypoints.

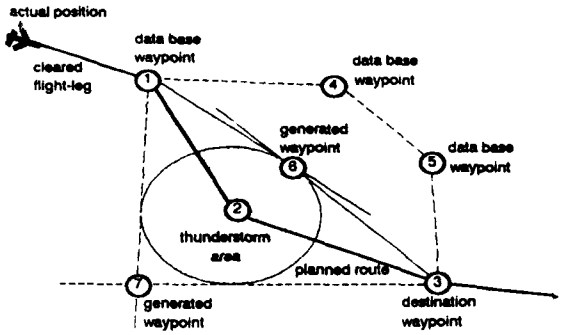


Figure 6 : Example for rerouting process

For the present state of implementation only one or two new waypoints are considered for each alternative. The evaluation will be expanded to any number of waypoints and new routes by improved algorithms. The application of these algorithms will be possible due to permanent increase in computing power. Eventually a complete regional flight will be automatically replanned in flight on the basis of knowledge about the destination, the actual situation and an extended navigational data base.

4. Experimental testing

For the experimental test phase the Cockpit Assistant is implemented in a flight simulator facility at the University of the Armed Forces in Munich. Therefore a one-seat fixed base cockpit with computer generated outside vision and head down display, artificial stick force as well as speech input and output has been used.

During the investigation nine professional pilots with a great amount of experience performed IFR approaches to Munich airport. For the evaluation, special scenarios have been simulated with and without the support of the Cockpit Assistant. These scenarios included special planning tasks under different conditions.

One major result was that all recommendations of the flight planning module were accepted. Besides this, the time to decision after the occurance of an event, which required replanning, could be significantly improved. Figure 7 represents the investigation results of the parameter "time to decision".

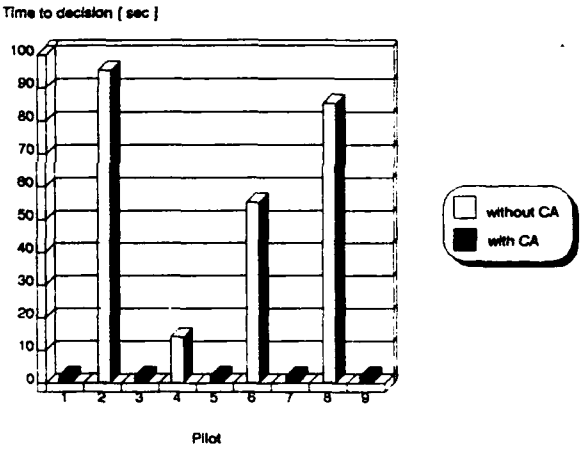


Figure 7 : Time to decision with and without Cockpit Assistant

5. Further development

These results, the significant improvements of flight accuracy and the good acceptance by the pilots [2] support future developments and extension of the assistance system functions. This will lead to an advanced Cockpit Assistant System called CASSY [11]. It will be developed for commuter aircraft in cooperation with Dornier.

According to the further development of the Cockpit Assistant the situation assessment and flight planning module will be improved and extended. As mentioned, the rerouting algorithms are presently revised for offering full route planning on the basis of a complete navigational data base, considering all accessable information about flight and system status and environmental conditions. Additional improvements of the evaluation criteria will conclude the lateral flight planning module.

The next step will be 3-D trajectory planning with regard to restricted areas and other altitude constraints from ATC. So, a conflict free lateral plan with the most important trajectory constraints will be generated. This can be the first step of integrating the system into the avionics of modern aircraft. The generated constraint list can possibly be fed into a Flight Management System to provide its normal functions. Today this is done by the pilot. However, the main task of the planning module is to provide the flightplan as input for other modules of the Cockpit Assistant.

A further step will be full 4-D trajectory planning as figure 8 shows, which represents the complete structure of the module.

For reasons of optimization the profile planning module is divided into two subtasks: Firstly, the tactical planning for economical flightlevel changes including step climbs and descends and secondly, the strategical planning of a complete trajectory using algorithms of the tactical profile planning module. The time planning has to meet the requirements of future Air Traffic Management and assure optimal 4-D planning.

Since CASSY, the advanced Cockpit Assistant, will include a dialogue manager to manage the communication between the crew and the system modules, it will be possible to offer more crew integration into the planning process. This improvement will even increase the acceptance of the pilots. Additionally, the situation assessment will be supported by another new module for recognizing when the pilot intentionally deviates from the flightplan.

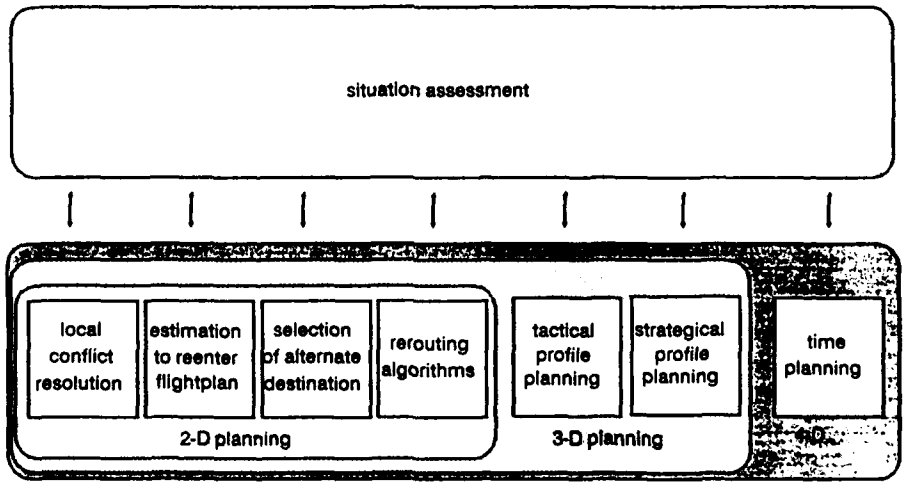


Figure 8 : Complete modular structure of the situation assessment and flight planning module

6. Concluding Remarks

The increasing complexity of aircrafts and Air Traffic Management require extensive assistance for the cockpit crews. Therefore, a Cockpit Assistant has been developed and a computer aid for flight planning tasks has been integrated.

Because of the results of the experimental investigation the further development of the system will be driven forward. Due to its overall structure and the big amount of included features CASSY could be a well qualified airborne assistant for crews, ATC, companies and their ground based computer facilities.

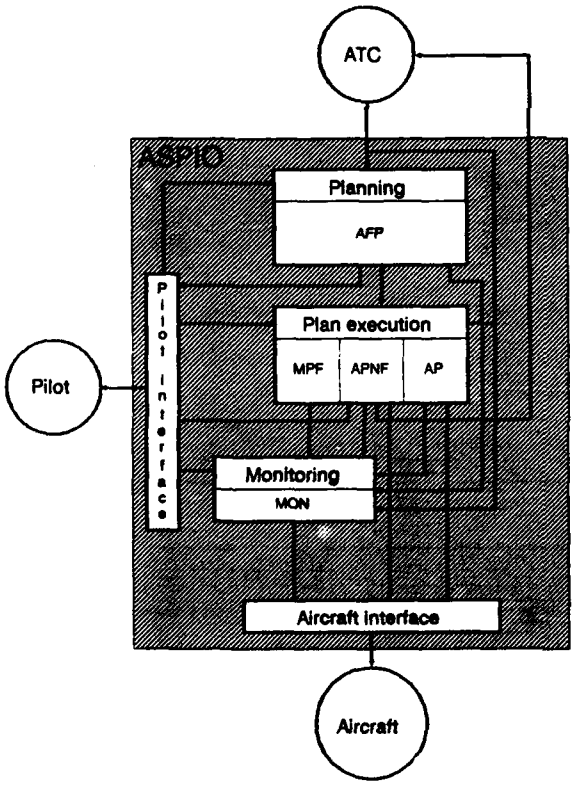


Figure 9 : Modular structure of the Assistant for Single Pilot IFR Operation ASPIO

References

[1] Harris, D.F.; Morrissette, J.A.
Single Pilot IFR Accident Data Analysis
NASA CR 3650, 1982.

[2] Rasmussen, J.
Skills, Rules and Knowledge; Signals, Signs and Symbols, and other Distinctions in Human Performance Models.
IEEE-SMC-13, No. 3, 1983.

[3] Dudek, H.-L.
Wissensbasierte Pilotenunterstützung im Ein-Mann-Cockpit bei Instrumentenflug
Dissertation, UniBw München, 1990.

[4] Onken, R.
Knowledge-Based Cockpit Assistant for IFR Operations.
AGARD GCP Symposium, Sept. 1990.

[5] Boy, G.
Intelligent Assistant Systems
Academic Press, London, 1991.

[6] Zadeh, L.A.
Fuzzy Sets
Information and Control, Vol. 8, 1965.

[7] Nilsson, N. J.
Principles of Artificial Intelligence
Springer Verlag, Berlin, 1982.

[8] Winston, P.H.
Artificial Intelligence
Addison-Wesley, Reading, Massachusetts, 1984.

[9] Rommelfanger, H.
Entscheiden bei Unschärfe
Springer-Verlag, Berlin, 1988.

[10] Kandel, A.
Fuzzy Mathematical Techniques with Applications
Addison Wesley, Reading, Massachusetts, 1986.

[11] Wittig, Th.; Onken, R.; Dudek, H.-L.;
Unterstützung des Piloten im IFR-Flug durch wissensbasiertes Assistenzsystem
DGLR Jahrestagung, Berlin, 1991.

Comparison of the Event-Step Algorithm to Other Path Planning Methods to Avoid Dynamic 3D Obstacles*

Mark Silbert
Code 7013
Naval Air Development Center
Warminster, PA 18974, USA

SUMMARY

In a previous paper [8], the Event-Step Algorithm (ESA) was presented as an efficient path planning algorithm to avoid dynamic obstacles. Dynamic obstacles are defined as obstacles that move and/or distort over time. This paper looks at the algorithm more closely and compares it to other path planning algorithms. Specifically, Grid Search, Voronoi Graph and Visibility Graph methods are considered. It shows that the ESA offers advantages over these methods and is a generalization and improvement of the Visibility Graph method. A summary of the ESA is presented.

1. INTRODUCTION

There is a strong desire to build computer-controlled robotic vehicles that can perform a variety of tasks. For example, robotic land vehicles can be used to penetrate hazardous or hostile landscapes to collect and/or disseminate data. A military application of this type would be to traverse the battlefield to perform bomb damage assessment. In a similar manner, robotic air vehicles could fly in and around the affected areas to collect pertinent data. In order for the vehicle to search the region unassisted by human operators, it needs to be able to sense its local environment to determine where it is, where it needs to go and what obstacles are in its way. Once the vehicle determines this, it then needs to generate a path that will get it from its current location to its destination while avoiding the obstacles in its way. This environment sensing and path planning are two distinct tasks that constitute route planning. In general, these two tasks must be done continually since there are always errors in sensing and the environment is likely to change over time. Developing methods to sense and interpret the environment is referred to as perception and is obviously a function of the sensor(s) used. Perception remains, to date, an extremely challenging problem that has not been solved. In fact, it is not even understood how perception by animals and humans is accomplished. Significant breakthroughs in sensor technology and sensor processing are critical to the success of this problem. These technologies are being developed under various other projects. Although sensing is a critical aspect of route planning, it will not be addressed in this paper. Instead, we will focus on the task of path planning by assuming that we are furnished with the vehicle's location, desired destination and the location, size and shape of all obstacles in the region. Additionally, we will assume that if any of the obstacles are moving, they are moving with constant speed and direction and we are provided this information as well. Clearly, these are big assumptions but they allow us to focus on what issues arise once this data is provided. We will show that even when all this information is provided, path planning is still a sophisticated and formidable problem. We will also show that the Event-Step Algorithm offers many advantages over the other methods in addressing the path planning problem with moving obstacles.

2. STATEMENT OF THE PROBLEM

Let us first define the path planning problem in general terms. Consider figure 2-1, which shows a two-dimensional example. The *planning region* is the area encompassing the vehicle's initial location, the desired destination and all obstacles considered. Assume the vehicle is located a point A and is trying to determine an *optimal path* to point B. The shaded regions are (stationary) obstacles in the region which must be avoided. The bold path in figure 2-1 is the optimal path for this problem. We define the optimal path, in this context, to mean the shortest distance path from the current position of the vehicle to the destination which avoids all obstacles. Thus, the problem is to devise an algorithm which will generate the optimal path which will get the vehicle to its desired destination while avoiding all potentially moving obstacles. In general, there may be multiple (possibly, infinite) optimal paths, in which case we assume that any one of them is a solution. Note that when all the obstacles are stationary, the problem is very similar to finding the optimal path through a maze.

3. GENERAL APPROACH TO PATH PLANNING

In general, all of the existing methods of path planning divide the problem into two phases. In the

* Rev 5, 12 Sep 91

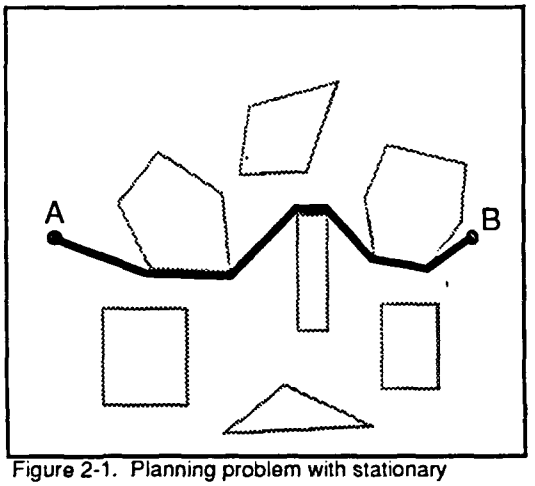


Figure 2-1. Planning problem with stationary obstacles. The bold line is the optimal path.

first phase, a (usually finite) search space is generated. This search space consists of all the possible paths that will be considered. This search space is represented as a graph. Keep in mind that the search space of the original problem is infinitely large since there are infinitely many ways of getting to the destination. Clearly, if an optimal path is to be found, then it must exist within the created search space. There are three common ways to build this search space: Voronoi graphs, Grids, and Visibility graphs. Once the search space is created, the second phase is to navigate through this space (i.e., graph) searching for an optimal path. The usual method of navigation is a form of best-first search. However, since a graph is being searched, uninformed methods such as depth-first and breadth-first search as well as partially-informed methods such as hill-climbing could also be used, although they would not, in general, be as efficient. The three main methods of generating the search space are discussed in this paper. A detailed discussion of hill-climbing, best-first, depth-first and breadth-first search can be found in [7, 10].

3.1 Voronoi Graphs

Voronoi graphs are made by placing a node at every "T" intersection of the corresponding Voronoi diagram. A "T" intersection is simply where three lines meet. The Voronoi diagram is the set of points that are equidistant from two or more obstacles. The (imaginary) borders of the operating region are treated as obstacles as well in constructing the Voronoi diagram. The resulting Voronoi diagram for figure 2-1 is shown in figure 3.1-1. The corresponding Voronoi graph is shown in figure 3.1-2. Note that the "T" intersections become nodes in the graph and the locus of points between them become edges. A more detailed discussion of Voronoi graphs is given in [4]. A nice feature of the Voronoi graph is that since there are only a few nodes generated, the entire graph may be rapidly searched. However, as pointed out, if the search space does not include the optimal path, then no amount of search will find it. In general, Voronoi graphs do not include an optimal path. Figure 3.1-3 shows the "optimal" path that would be generated for the problem in figure 2-1. Comparing the paths, we see there are many needless "kinks" in the Voronoi paths. Worse yet, since the Voronoi graph is a function of the distance between the obstacles, progressively inferior paths result when the obstacles are further apart. This problem can easily be seen by drawing the Voronoi diagram for a single obstacle as shown in figure 3.1-4. This figure shows an obstacle that is barely on the direct path from A to B. However, instead of steering the vehicle slightly off the direct path to avoid the obstacle, it directs the vehicle along the much longer paths that are equidistant between the obstacle and the planning region borders.

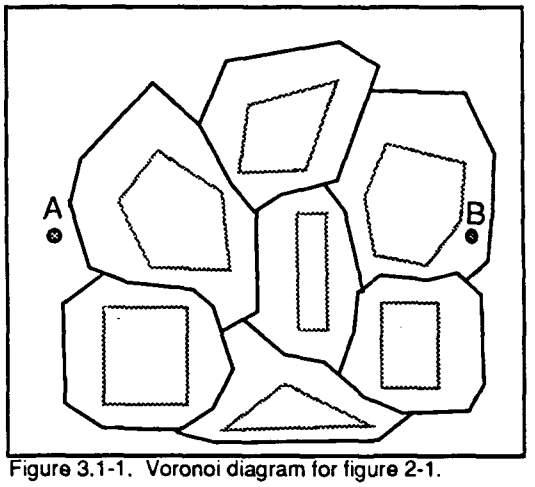


Figure 3.1-1. Voronoi diagram for figure 2-1.

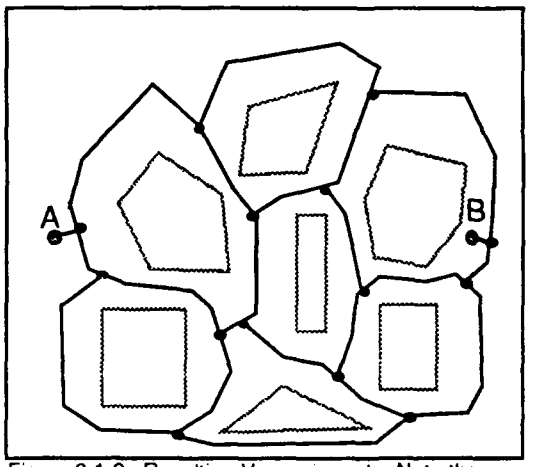


Figure 3.1-2. Resulting Voronoi graph. Note the nodes of the graph occur at the "T" intersections.

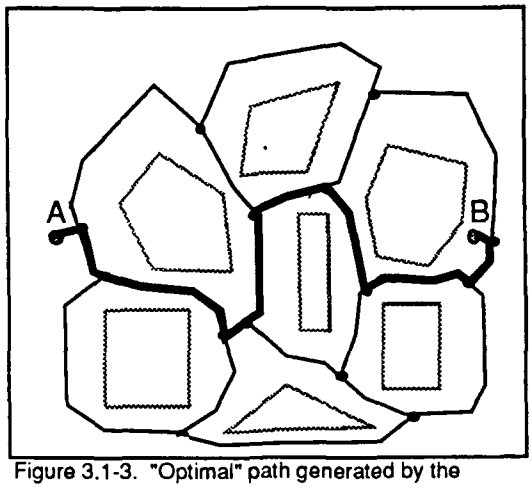


Figure 3.1-3. "Optimal" path generated by the Voronoi graph. Note the path has substantially more kinks than the optimal path in figure 2-1.

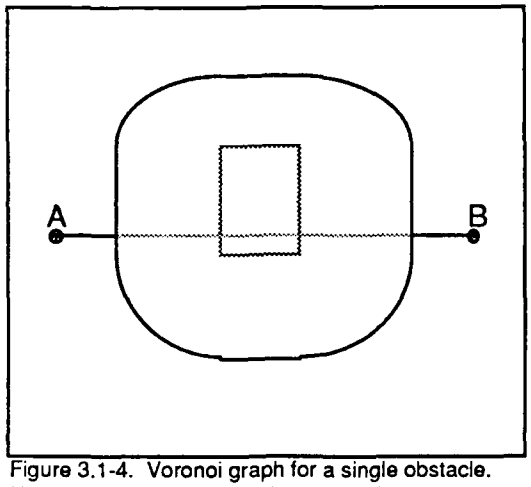


Figure 3.1-4. Voronoi graph for a single obstacle. Note that the path guides the vehicle far from the direct path even though the obstacle is small and is barely in its path.

Based on the discussion of Voronoi graphs, one can see that the attempts to extend this method to either three-dimensional worlds or moving obstacles are futile. In three-dimensional worlds, the set of points equidistant between two or more obstacles becomes planes instead of lines. Thus there become infinitely many ways to generate paths between any two obstacles. Moving obstacles are not handled either because the set of points making up the Voronoi diagram is changing over time and thus the corresponding Voronoi graph is changing as well.

Thus we see that path planning algorithms based on Voronoi graphs not only do not extend to three-dimensional or moving obstacle problems, they do not even produce optimal paths for two-dimensional problems with static obstacles.

3.2 Grids

Grids are made by overlaying an array of cells over the planning region as shown in figure 3.2-1. Cells that are more than half filled by an underlying obstacle are marked as impassable. Paths are then found by moving through adjacent cells of the grid that are passable until the cell containing the destination is reached. Note that the cells become the nodes of the graph and the cell-to-cell transitions are the edges. The number of cells used in each dimension is a parameter but a trade-off in time versus storage and path quality emerges. Namely, as the number of cells increases there is more opportunity of including the optimal path within the search space. However, more storage is needed to maintain the grid and more time is required to find the path since the search space is larger.

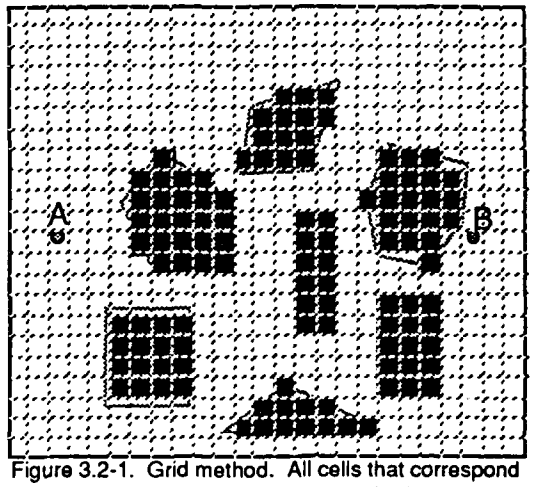


Figure 3.2-1. Grid method. All cells that correspond to the location of an obstacle are marked as impassable.

Figure 3.2-2 shows the best path in the search space. Comparing it to figure 2-1 we see that the path created using the grid method is similar but a bit more jagged. As the number of cells increases, a smoother path, asymptotically approaching the optimal path in figure 2-1, would emerge. In figure 3.2-3 we again use the grid method but with a larger cell size. Note that previously passable gaps between the obstacles are now lost and the best path in the resulting search space is quite inferior to the optimal path in the original problem.

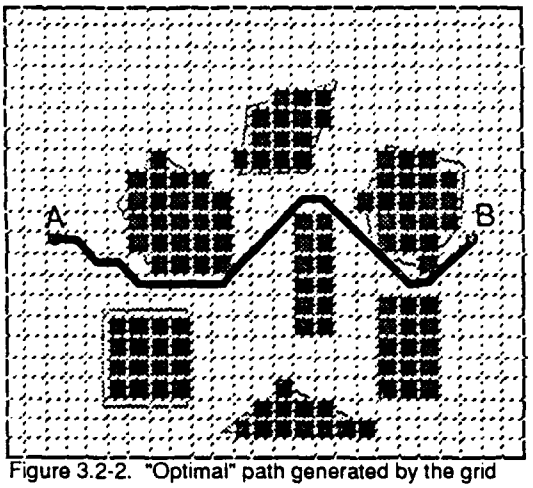


Figure 3.2-2. "Optimal" path generated by the grid method. Similar to the optimal path in figure 2-1 but more jagged.

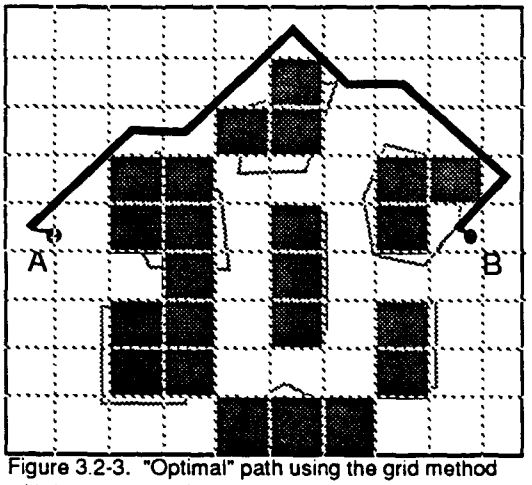


Figure 3.2-3. "Optimal" path using the grid method with larger cells. Note the substantially inferior path.

Applying the grid method to three-dimensional problems involves the use of three-dimensional grids. All other aspects of the method remain the same. Of course when three-dimensional problems are addressed, the number of cells drastically increases and thus time versus path quality becomes even more of an issue.

In a similar manner, once the obstacles are allowed to move, we need to build a sequence of grids representing the position of the obstacles at various points in time. Thus, two-dimensional problems require three-dimensional arrays and three-dimensional problems require four-dimensional arrays!

As a result, path planning algorithms based on the grid method require both extremely large memories and high processing capability to obtain fairly optimal paths in a reasonable time.

3.3 Visibility Graphs

Visibility graphs are made by establishing the straightline visibility between the vertices of all obstacles along with the vehicle's initial and desired locations. To form this graph, the obstacles are modeled as a set of convex polygons. Then, the set containing all of the obstacles' vertices along with the vehicle's initial and desired locations become the nodes of the visibility graph. The edges of the visibility graph are made by determining all the straight, uninterrupted paths between these nodes. An uninterrupted path is one that doesn't cross or clip any obstacle. The visibility graph for figure 2-1 is shown in figure 3.3-1. Note that the optimal path is contained within the visibility graph. This was not coincidental. In fact, it can been proven that the optimal path for two-dimensional problems is always contained within the visibility graph. The proof is based on imagining tying a rubber band between the nodes A and B. The rubber band is then stretched by "threading" it around the obstacles until it lays flat without overlapping any obstacle. It should be intuitively clear that any optimal "threading" of the rubber band will coincide with the edges of the visibility graph. Generation of the optimal path, along with the fact that visibility graphs are reasonably small, makes them a very attractive method for solving two-dimensional (static) obstacle avoidance problems.

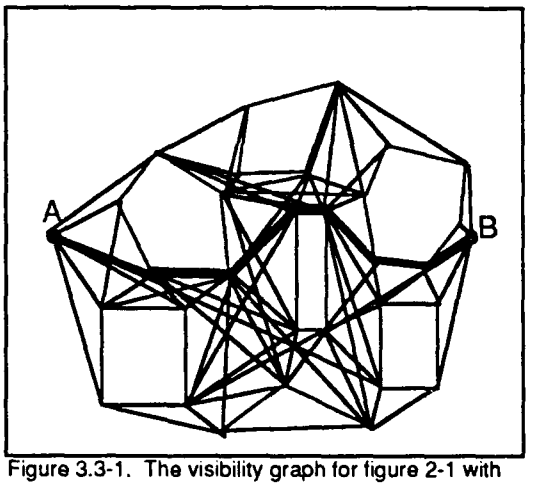


Figure 3.3-1. The visibility graph for figure 2-1 with its optimal path. This optimal path is identical to the path in figure 2-1.

However, things get deceptively hard when one attempts to extend visibility graphs to either three-dimensional or moving obstacle problems. At first, it would seem that visibility graphs could handle three-dimensional obstacles by simply treating them

as polyhedrons, using all its vertices as the nodes and all the uninterrupted paths between them as edges. Unfortunately, the optimal path would not, in general, be contained in this graph. As an example, consider the problem where the vehicle is sitting on top of a large rectangular block and is trying to find the optimal path to the floor in front of the vehicle. The optimal path is to simply go straight ahead to the edge of the block and then move down to the floor. However, this path would not be included in the visibility graph. Instead, the vehicle would have to first move to one of the upper corners of the block, then down to the floor, and then move back to the desired spot on the floor. As a result, three-dimensional problems are not handled by the visibility graph method.

Visibility graphs do not handle moving obstacles for a similar reason that Voronoi graphs do not. Since the visibility graph changes over time as the obstacles move, one cannot build a static graph and expect it to be useful in a dynamic environment.

So we see that while visibility graphs are an excellent choice for planning problems where the obstacles are static and two-dimensional, they are deficient when the obstacles are moving and/or three-dimensional.

4. THE EVENT-STEP ALGORITHM

It will be shown that the Event-Step Algorithm (ESA) is a generalization and improvement of the visibility graph method. Namely, the ESA allows dynamic and/or three-dimensional obstacles but when the obstacles are two-dimensional and static, the ESA reduces to the visibility graph method, except it builds a smaller (implicit) graph. In addition, we claim that this smaller graph contains the optimal path. Although we do NOT claim to produce the optimal path for moving obstacle problems, the paths generated are of high quality. In actuality, finding the optimal path for dynamic obstacle problems has been shown to be NP-Hard. However, the ESA demonstrates that by forgoing optimality, it can generate nearly optimal paths for dynamic obstacle problems in polynomial time. In fact, it will be shown that the ESA can handle dynamic obstacles without introducing significantly more time complexity than the static obstacles case.

Details of the ESA are given in Silbert [8], but are summarized here. One of the key aspects of the ESA is that the search space graph is built implicitly, as needed, instead of built all at once, statically. The part of the graph that is required is determined by computing intercepts. To explain the ESA, let us use the two-dimensional example shown in figure 4-1. The figure shows the initial vehicle location, the desired destination and an intervening obstacle. Assume, for now, that the obstacle is stationary.

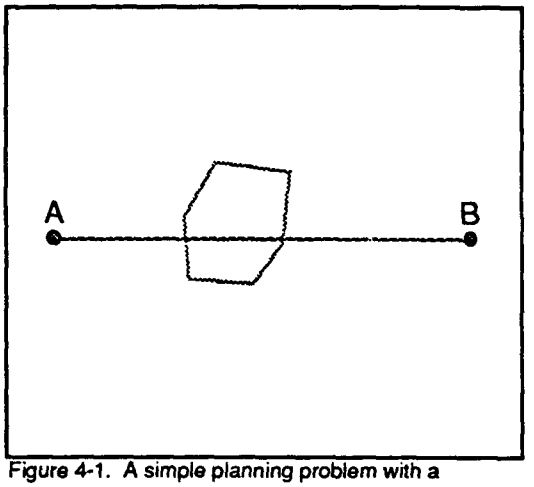


Figure 4-1. A simple planning problem with a stationary obstacle.

Like the visibility graph method, all obstacles are modeled as convex polygons.[1] The algorithm first checks to see if any obstacles, i.e., polygons, are in the direct path to the desired location. For the given example, the obstacle is on this direct path. The ESA then computes the clockwise and counter-clockwise paths around the obstacle. To go clockwise around the obstacle, the most counter-clockwise vertex of the polygon is found and used as the new (intermediate) destination (see figure 4-2). This counter-clockwise vertex is the one which yields the most left-hand turn away from the direct path to the destination.

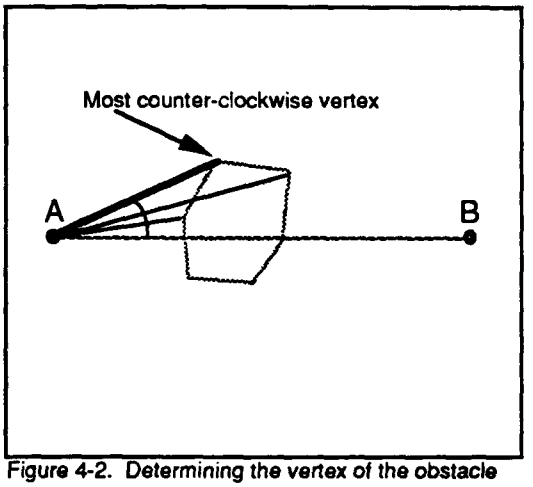


Figure 4-2. Determining the vertex of the obstacle that yields the most counter-clockwise turn from the direct path to B.

[1] As pointed out in [8], requiring all polygons to be convex is not a real restriction since any arbitrary (e.g., concave) polygon can be represented as a set of one or more convex polygons.

Then by constructing the line from this vertex to the final destination, a determination is made whether to move to the next clockwise vertex or to move directly to the final destination (figure 4-3). This determination is made by simply choosing the one which yields the lesser clockwise turn. If the final destination requires the lesser turn, we then have the clockwise path around the obstacle. If the next clockwise vertex yields the lesser turn, we move to that vertex and repeat the process. The final clockwise path around the obstacle is shown in figure 4-4.

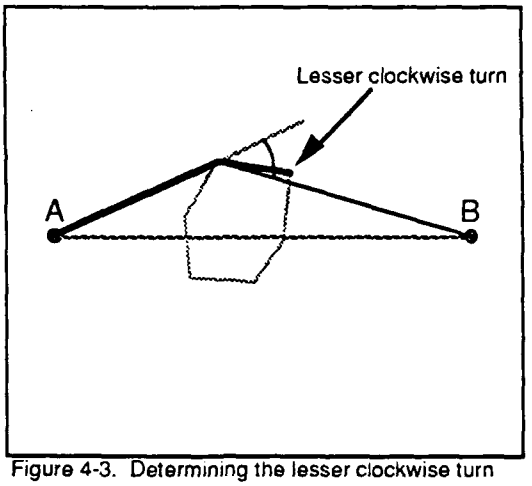


Figure 4-3. Determining the lesser clockwise turn around the obstacle.

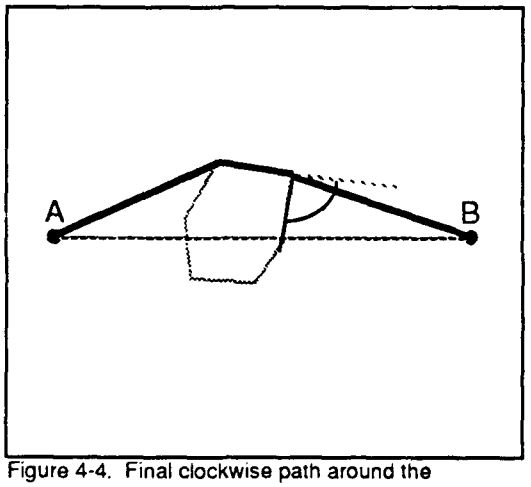


Figure 4-4. Final clockwise path around the obstacle.

Going counter-clockwise around the obstacle is handled similarly. Namely, as shown in figure 4-5, the most clockwise vertex is found and then we choose the lesser counter-clockwise vertex each time. The counter-clockwise path is shown in figure 4-6. If this were the only obstacle, the ESA would then simply compare the distances of these two paths and choose the smaller one, yielding the same optimal path as the visibility graph method.

Multiple obstacles are handled by recursively finding these clockwise and counter-clockwise paths for each obstacle as it is encountered. Namely, the ESA finds the clockwise path around the first obstacle encountered. As it generates this clockwise path, it checks to see if any other obstacles are encountered. If an obstacle is encountered, the ESA recursively applies itself to solve this subproblem. Similar processing then occurs for the counter-clockwise path around the original obstacle. As pointed out in [8], simply applying this obstacle avoidance method to each subproblem can lead to inefficiencies in the overall solution. Also in [8], this problem is remedied by building and evaluating the extra paths that are made by including the *maximal deviate points*. The maximal deviate points are those points that deviate the furthest from the direct path to the destination.

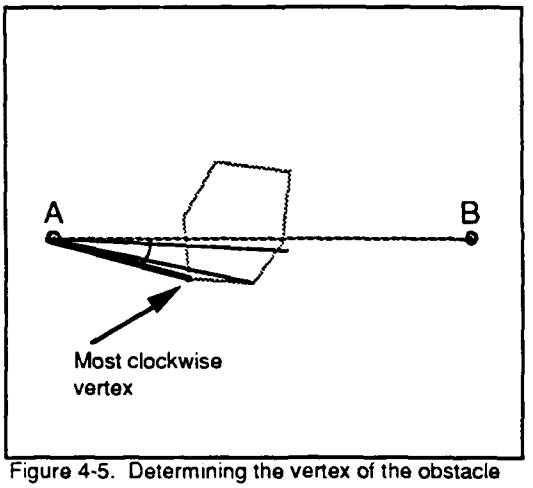


Figure 4-5. Determining the vertex of the obstacle yielding the most clockwise turn from the direct path to B.

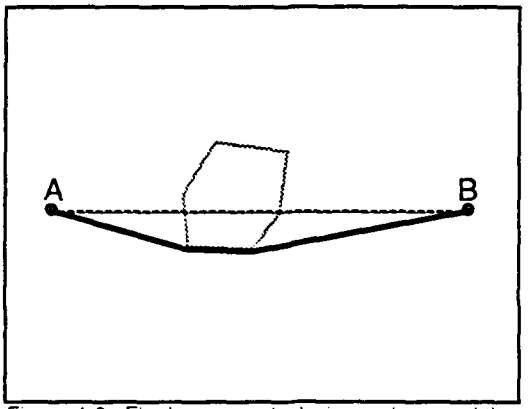


Figure 4-6. Final counter-clockwise path around the obstacle.

Figure 4-7 shows the result of applying the ESA to the original problem in figure 2-1. The dotted paths are the ones that were built and evaluated prior to selecting the optimal path. Note that the dotted lines make up a subset of the visibility graph. Namely, in general, there are two classes of lines that occur in the visibility graph but are not generated by the ESA: 1) lines that are not *locally tangent* to the obstacles and 2) lines to and between obstacles that are so far off the direct path that they need not be considered. A line is locally tangent to two or more polygons if it intercepts at least one vertex of each polygon but does not penetrate it (i.e., does not pass through the interior of the polygon). Since the optimal path includes neither of these line classes, we are sure that the optimal path is not discarded by the ESA. In other words, *the ESA will generate and return the optimal path for static two-dimensional problems without generating all the nodes and edges of the visibility graph.*

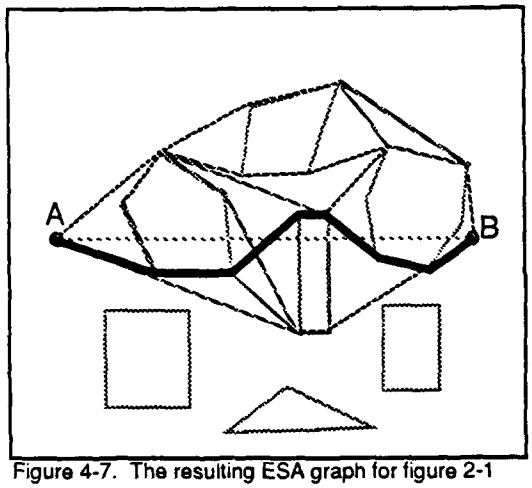


Figure 4-7. The resulting ESA graph for figure 2-1 with its optimal path. Note the reduction in the number of edges generated relative to the visibility graph in figure 3.3-1. Like the visibility graph, however, the optimal path is identical to the path in figure 2-1.

Moving obstacles are handled by the ESA in an analogous way except each vertex of the polygon is assumed to be moving with constant speed and direction. Therefore, given the speed of the vehicle, the location of each vertex polygon intercept can be determined. The time at which the vehicle arrives at the moving vertex is computed as:

$$(s \cdot t)^2 = (x_i + v_x \cdot t - x)^2 + (y_i + v_y \cdot t - y)^2$$

where x_i, y_i is the ith vertex of the polygon,
v_x, v_y are the X and Y components of the polygon's velocity vector
x,y is the location of the vehicle
s is the speed of the vehicle
and t is the computed time when the vehicle arrives at the polygon's ith vertex.

Note that this equation is based on the Pythagorean Theorem for right triangles. Using this equation, the ESA determines the clockwise and counter-clockwise paths around the moving obstacle as before. In figures 4-8 and 4-9, we again have a single obstacle except this time it is moving in a southeasterly direction. Figure 4-8 shows the resulting clockwise path around the obstacle. Figure 4-9 is the resulting counter-clockwise path.

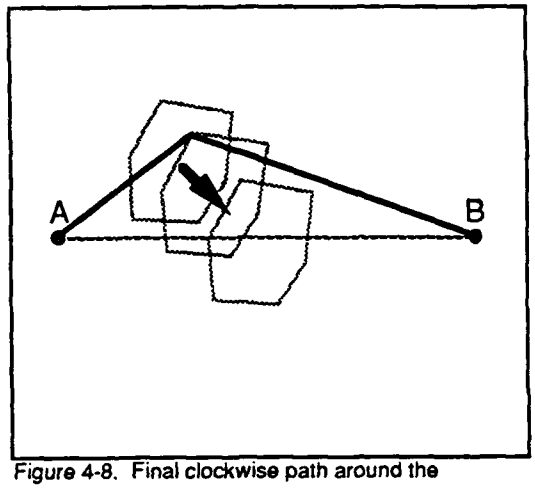


Figure 4-8. Final clockwise path around the moving obstacle.

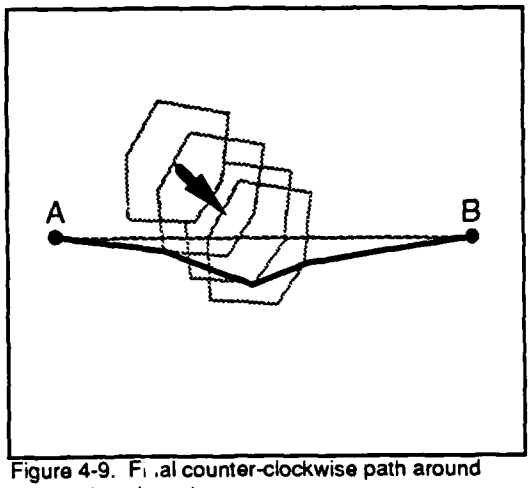


Figure 4-9. Final counter-clockwise path around the moving obstacle.

The ESA can also be extended to a special case of three-dimensional obstacles. These obstacles are ones that have some arbitrary polygonal cross-section (as already assumed) but have a specified lower and upper height. This transforms the obstacle model from a set of convex polygons to a set of generalized convex "cylinders". Using these generalized cylinders as the three-dimensional model for obstacles, we see there

are four locally minimal ways to go around it: clockwise in the horizontal and vertical planes and counter-clockwise in the horizontal and vertical planes. The clockwise and counter-clockwise paths in the horizontal are the same two paths found for the two-dimensional problems. One can think of the path that goes clockwise in the vertical plane as going over the obstacle and counter-clockwise in the vertical plane as going under the obstacle. Thus, the same strategy used to move around the obstacles is used to move over and under them as well. Note that, unlike the visibility graph method, the strategy used by the ESA would correctly solve the problem presented earlier where the vehicle is sitting on a block and is trying to get to the floor directly in front of it. The ESA does not strictly depend on the vertices the way the visibility graph method does. It only considers clockwise and counter-clockwise movement around the obstacles. Also note that it is important to use only the special case of three-dimensional obstacles previously defined since only they have the property of having four locally minimal paths around it. Namely, once slopes and/or curves are included in the obstacle model, there may be many locally minimal paths around them. A worst case example is modelling an obstacle as a sphere which has an infinite number of locally minimal paths around it.

5. ANALYSIS OF THE ESA

Since the ESA has been shown to be an extension and improvement over the visibility graph methods, it makes it easier to quantify its performance. The size of the visibility graph is a function of the number of obstacles and the number of vertices per obstacle. The analysis will make use of the standard big "O" notation found in [1]. For simplicity, assume there are N obstacles each containing M vertices. Therefore, there are N•M + 2 nodes in the visibility graph (Note the initial and desired locations of the vehicle are treated as nodes). Thus, in worst case there are (1/2)•((N•M + 2)^2 - (N•M + 2)) edges in the graph. Since the time complexity is directly proportional to the number of edges in the graph, the complexity is O((N•M)^2). Thus the worst case time complexity for the ESA for static, two-dimensional problems is O((N•M)^2). However, since the ESA does not build the entire visibility graph, the actual number of edges is much smaller. When the ESA is applied to moving obstacle problems, we see that there are really no extra edges generated and the new position of the node is computed in constant time. Thus, this is the worst case time complexity for all (i.e., both static and dynamic) two-dimensional problems. Note how this contrasts against the other methods presented. Recall that the Voronoi and visibility graph methods simply don't handle moving obstacles while the grid method requires significantly more memory and processing.

When dealing with three-dimensional obstacles, a similar result occurs. The ESA needs to generate the extra over and under the obstacle edges but characterizing the added complexity becomes a bit unclear. To go over these generalized cylinders requires two nodes. Similarly, going under also requires two nodes. Thus we see that four extra nodes are effectively added to each obstacle. Since this is a constant, we can ignore these extra nodes in the order of time complexity calculation. Thus we see that the time complexity to handle three-dimensional obstacles remains O((N•M)^2). Therefore, we still see substantial benefit of the ESA to the other methods mentioned. Recall that the Voronoi and the visibility graph methods do not extend to three-dimensional problems and the grid method requires one to use substantially more memory and processing.

6. CONCLUSION

The ESA is a powerful path planning algorithm that requires only moderate memory size and processing power to execute. Furthermore, when the problem involves static, two-dimensional problems, the ESA will find the optimal path. When the problem involves dynamic obstacles, the guarantee of optimality is lost but the ESA easily generalizes to these problems and requires negligibly more processing to solve them. It should be noted that finding the optimal path for dynamic obstacle problems has been shown to be NP-Hard. Thus, the ESA, like any other polynomial algorithm can not possibly find the optimal solution for all geometries. However, the paths found by the ESA are of high quality and are found quickly. Thus the ESA forgoes optimality but generates nearly optimal paths in polynomial time. Finally, a special case of three-dimensional obstacle avoidance problems can also be handled by the ESA without much more processing requirements.

REFERENCES

1. Aho, A., Hopcroft, J., and Ullman, J., "The Design and Analysis of Computer Algorithms", *Addison-Wesley Publishing* (1974)

2. Beaton, R., Adams, M., and Harrison, J., "Real-Time Mission and Trajectory Planning", *Proceedings of the 26th IEEE Conference on Decision and Control* (Dec 1987), pp. 1954-1959

3. Lozano-Perez, T., and Wesley, M., "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles", *Communications of the ACM* 22,10 (Oct 1979), pp. 560-570

4. Meng, A., "Free Space Modelling and Geometric Motion Planning", *IEEE* (Aug 1988), pp. 82-87

5. Meng, A., Ntafos, S., and Tsoukalas, M., "An Approach to Real-Time Path Planning for Handling Changing Targets and Unexpected Threats", *Proceedings of Associate Technology Conference* (1991), pp. 184-192

6. Mitchell, J., "An Algorithmic Approach to Some Problems in Terrain Navigation", *Artificial Intelligence Journal* (Dec 1988), Vol. 37, Numbers 1-3, pp. 171-201

7. Nilsson, N., "Principles of Artificial Intelligence", *Tioga Publishing* (1980)

8. Silbert, M., "An Efficient Algorithm for Path Planning to Avoid Dynamic 3D Obstacles", *Proceedings of Associate Technology Conference* (1991), pp. 193-204

9. Tychonievich, L., et al. "A Maneuvering-Board Approach to Path Planning with Moving Obstacles", *Eleventh International Joint Conference on Artificial Intelligence* (Aug 1989), Vol. 2, pp. 1017-1021

10. Winston, P., "Artificial Intelligence (Second Edition)", *Addison Wesley* (1984)

On Board Planning of 4D-Trajectories

V.Adam
R.Kohrs
Institute of Flight Guidance
German Aerospace Research Establishment (DLR)
Postfach 3267
D 3300 Braunschweig, FRG

1. SUMMARY

The evolution of present day Air Traffic Management (ATM) towards an integrated Air Traffic Management will lead to the introduction of computer based planning systems on the ground side, which communicate with advanced Flight Management Systems onboard the aircraft via automatic data link. An experimental Flight Management System (EFMS) is under development in the frame of the PHARE program for research into future ATM concepts. This EFMS will have the capability to predict a 4-dimensional trajectory according to a set of constraints given by an ATC planning computer on the ground. It will negotiate this trajectory with ATC via data link and will generate appropriate guidance commands for the autopilot/autothrottle system to steer the aircraft along this trajectory. This paper describes briefly the functionality of the EFMS and then concentrates on the planning strategy for the generation of a trajectory from the departure to the destination airport which meets the defined constraint attributes.

2. INTRODUCTION

Increasing numbers of transport aircraft are being supplied with or else retrofitted with Flight Management Systems (FMS). Their installation offers aircraft operators the potential for cost savings by providing guidance to trajectories optimised to their operating criteria. However, in many parts of Europe, the potential advantages are only partially realised due to external constraints placed upon aircraft speed/height trajectories by the Air Traffic Services. Such constraints normally result because the density of Air Traffic prevents ATC offering aircraft their desired flight trajectories. As the density of traffic increases towards the start of the 21st Century, these constraints are very likely to become more severe.

The opportunity exists for significant improvement in capacity and efficiency of the present day Air Traffic Management (ATM) system by the much closer integration of airborne and ground based computer systems.

The picture emerging is of a future ATM system using the predicted flight trajectory information of participating aircraft, to detect and resolve conflicts over substantially greater time horizons than is currently the case. A process of negotiation might take place between ground and airborne systems to establish a 3- or 4-dimensional flight path to be flown. The trajectory might conceivably be expressed as a 4-dimensional volume of airspace (a 'bubble' moving in time), within which the aircraft would be required to remain. The exact dimensions might depend on factors such as traffic density at that time. This could allow aircraft to optimise their exact trajectories while being flown, thus taking into account changes in meteorological conditions or mission priorities.

Such a concept requires the development of many techniques not currently in operational use within the Air Transport Industry. Examples include flight trajectory data exchange using evolving air-ground data links, 4-dimensional trajectory prediction and guidance, conflict detection and resolution techniques to mention only a few.

Eurocontrol Member States have agreed to develop the means for executing research on these topics under the umbrella of the 'Programme for Harmonised ATM Research in EUROCONTROL' (PHARE). This initiative will effectively combine complementary expertise available in the member organisations. The ultimate objective will be a full demonstration of an ATM system working under the concepts developed in the PHARE program, involving ground ATC simulation with simulated, FMS equipped aircraft followed by trials employing fully equipped research aircraft flying in a simulated ATC environment.

3. EXPERIMENTAL FLIGHT MANAGEMENT SYSTEM

A common requirement of the partners in the PHARE program is an Experimental Flight Management System (EFMS) [1], [2], that will serve as a research tool to allow development and evaluation of techniques and functions mentioned above. The EFMS will not require the complexity and integrity of full commercial equipment and many functions, such as sensor management, are entirely superfluous. However, two issues are crucial. Firstly researchers must, in the light of their experiments, be able to develop and modify algorithms incorporated in the EFMS. Secondly, complementary research activities will need to exchange the products of their work in the form of algorithms or software modules.

3.1 Interfaces of the EFMS

The EFMS provides interfaces to the outside world, mainly pilot, experimenter, ATC and aircraft, as shown in Figure 1.

The pilot communicates with the EFMS via a Control and Display Unit (CDU), which will be utilised to enter performance parameters or to assemble a list of constraints defining the route of flight. The Electronic Flight Instrument System (EFIS) will serve for the graphical representation of the predicted flight trajectory.

Before running an experiment the experimenter (Supervisor) may upload preparation files and modify certain experimental

parameters or select different recording options via a terminal.

The ATC interface connects the EFMS to an automatic data link. There will be an automatic exchange of weather data. Onboard measured wind, temperature and position data will be downlinked to update a weather data base on the ground which may be used by the EFMS to query the latest weather data.

The aircraft's prefered trajectory planned to chosen mission objectives could be negotiated with the ATC planning computer. ATC may send a list of constraints at certain waypoints, which could allow the respective aircraft to be safely and efficiently merged into a dense flow of traffic.

The EFMS receives the aircraft's 'State Vectors' from the sensor system and delivers the 'Outer Loop Guidance Vector' to the autopilot/autothrottle system.

3.2 Major Processes of the EFMS

The internal structure of the EFMS is depicted in Figure 2 in a simplified manner. All internal processes are represented by 'bubbles'. Data stores are shown as rectangular boxes. Directed lines between processes, data stores and the external interfaces indicate the flow of data.

Process 1 (Assemble Constraint List) assembles a list of constraints in chronological order. A constraint consists of a waypoint with additional attributes. For the construction of a constraint list the pilot may select from the navigation data base a route or a part of a route by name or as a series of waypoints. Constraints may also be send from ATC via data link.

Process 2 (Manage Performance Parameters) generates a list of performance parameters which forms the 'Phase Table'. These performance parameters (thrust settings, climb speeds, cruise altitude etc.) could either be manually entered by the pilot via the CDU or could be called up by the pilot from a 'Performance Handbook' contained in the data base of the EFMS.

Process 3 (Manage Meteo) retrieves meteo data (wind and air temperature) along the intended route from a data store providing a meteo grid of the respective area. In the Phase 1 version of the EFMS the meteo grid will be uploaded by the experimenter by means of a preperation file prior to the experiment. In a later version it will be queried from the meteo data base on the ground via data link.

Process 4 (Predict Trajectory) generates the 'Proposed Trajectory' composed of the lateral route and the vertical profile. Major inputs to this process are a list of constraints and a list of performance parameters entered by the pilot as well as meteo information along the route. The state vectors are used to define the initial planning conditions (aircraft airborne or on the ground). The prediction of the lateral route is based on the sequence of waypoints contained in the constraint list. The vertical profile is made up of a series of flight phases defined by the 'Phase Table' set up by the pilot.

Process 5 (Negotiate Contract) sends a description of the 'Proposed Trajectory', generated by process 4, down to ATC, if the pilot requires a clearance for that trajectory. The process receives a tube description defined by ATC, which is based upon the requested trajectory giving 4D tolerances around the trajectory that provide a margin to allow for flight technical error, meteo changes, performance variations, etc.. Trajectory and tube description are also made available for display to the pilot. Trajectory and tube are then activated, if the pilot agrees with the tube assigned by ATC.

Process 6 (Guide Aircraft through Tube) generates guidance commands taking the 'Active Trajectory' and the aircraft state as inputs. These guidance commands are then fed into the autopilot/autothrottle system to steer the aircraft along the trajectory. The position of the aircraft within the 'Active Tube' is monitored and a warning is issued to the pilot, if the aircraft's flight path attempts to infringe the tube.

4. PREDICTION OF 4D-TRAJECTORIES

A major function of the EFMS is to generate a 4D trajectory from the departure airport to the destination airport which meets the defined constraints. In the lateral plane, this trajectory will be along great circle arcs between waypoints and will follow circular arcs of defined radii when turning from one leg to the next. The generation of altitude and airspeed profile is based on generic profiles, which result from the application of the initial 'Phase Table' together with initial conditions (flight status, aircraft initial weight etc.). The profile will be modified to fit within the constraints by changing altitude, speed, thrust index and energy sharing index initially defined in the 'Phase Table' in an appropriate manner.

The prediction of the trajectory comprises:

- generation of route description (2D-trajectory) for the given set of waypoints contained in the constraint list
- generation of altitude (3D-trajectory) and airspeed profile according to the given performance parameters
- calculation of ground speed, range and flight time (4D-trajectory) taking wind into account
- modification of 4D-trajectory to meet all constraints

The prediction of the horizontal path makes use of vector algebra. The prediction of the airspeed and altitude profiles is based on the integration of the equations of motion, which employ models describing

- lift and drag of the aircraft,
- net engine thrust and fuel flow rate,
- properties of the standard atmosphere.

The strategy of how to predict the complete trajectory, including several measures to modify the trajectory such that all constraints are met, is described in the following chapters.

4.1 Prediction of Horizontal Route

The horizontal route between departure airport or present position and final constraint point consists of great circle arc segments and circular turn segments at certain waypoints. An intercept path, a holding pattern or a stretched fan path may be included, as is shown in Figure 3. Three different types of waypoints are foreseen to comply with complicated route structures:

Type 1: **Regular** waypoints, which describe the route without the turns when directly connected

Type 2: **Centre of Turn** points, which could be used to construct more complicated approach paths including turns for more than 180 degrees

Type 3: **End of Turn** or **Start of Turn** points, which allow to predict an intercept path, a fan path or a holding pattern

The prediction starts at the first waypoint of the route (departure airport or present position) and runs to the last waypoint of the route (destination airport or designated final waypoint).

4.2 Prediction of Vertical Profile

The prediction of the vertical profile comprises up to three major phases: climb, cruise and descent, which are each composed of several subphases as shown in Figure 4. Depending on the present status of the aircraft (on ground or airborne) the planning strategy is selected. This defines the start of the predicted trajectory as either the take off point on the departure runway or as the present position in flight.

If the aircraft is still on the ground the prediction of the trajectory starts with a climb phase, proceeds with a descent phase and finally fits in the cruise phase. If the aircraft is already flying the prediction of the trajectory starts with a 40 seconds straight and level flight segment. From there follows a climb, cruise or descent segment depending on the present state of the aircraft, the length of the route and other parameters.

Altitude and airspeed profile are predicted by integration of the equations of motion for each subphase according to thrust mode, thrust index as well as subphase target values for altitude, airspeed and energy sharing index as described in the 'Phase Table' (see Figure 5).

The predicted flight time along the trajectory is received from integration of time and ground speed (on great circle straight segments) or turn rate (on constant radius turn segments) respectively until reaching a certain trajectory point (e.g. start of subphase, start of turn, middle of turn, end of turn etc.) on the trajectory.

The climb phase and the cruise acceleration phase are predicted employing a commanded thrust index setting. The first subphase is flown at take off thrust and constant speed CAS_CL0 until reaching the thrust reduction altitude, where the thrust is reduced to maximum continuous thrust. The third climb subphase is performed as an energy sharing climb, where the energy sharing index defines the proportion between climbing and accelerating. The following climb subphases are performed either at constant speed (CAS_CL1, CAS_CL2, MACH_CL) or at constant altitude (FL100) to accelerate the aircraft from CAS_CL1 to CAS_CL2. This part of the profile prediction is always performed by forward integration until reaching either the Top of Climb (TOC) or the Cruise Acceleration Point (CAP) at cruise altitude ALT_CR, as shown in Figure 4.

At this point the final weight is estimated from the predicted aircraft weight and then used for the subsequent prediction of the descent profile.

The whole descent phase and the deceleration subphase at cruise altitude are predicted by backward integration of the equations of motion for each subphase according to the thrust index and subphase target values for altitude, airspeed and energy sharing index stored in the 'Phase Table'. The prediction of the descent profile starts at the Gate taking altitude ALT_GT, speed CAS_GT and estimated landing weight as initial values and runs until the Cruise Deceleration Point (CDP) is reached.

The descent profile provides two level segments, one before reaching the Gate and a second at an intermediate altitude, e.g. FL100 or assigned altitude at the Metering Fix to provide some margin for the possible shift of the position of the end point of the idle descent and deceleration subphase. Major portions of the descent are performed at constant speed (MACH_DE, CAS_DE1, CAS_DE2).

Subsequently the cruise phase is predicted by forward integration of the equations of motion starting at the Top of Climb until reaching the Cruise Deceleration Point, which was found in the course of the prediction of the descent profile.

The aircraft weights at CDP received from forward cruise prediction and backward descent prediction are utilised to revise the estimated final weight. Depending on the difference in final weights the prediction of the descent and cruise profile may be repeated.

4.3 Strategies to meet Constraints

Constraints are one of the major inputs to trajectory prediction. Constraints are windows in space and time through which the aircraft must fly. A constraint consists of a waypoint, together with a constraint type, describing the nature of the constraint such as turning point, height constraint point, speed constraint point or time constraint point. A constraint will also have attributes, detailing the actual limitations being placed upon the aircraft at that constraint, e.g. size of airspace allocated or time slot that the aircraft must meet.

The development of strategies for the prediction of trajectories, which comply with multiple 4D constraints is a major research issue. Many parameters and several criteria have to be considered, such as

- time accuracy of predicted trajectory,
- efficiency of predicted trajectory as regards fuel consumption
- smoothness of thrust control,
- acceptance by pilots,
- computation time,
- number of iterations needed

to mention only a few. The following chapters describe some initial solutions of how to comply with horizontal, airspeed, altitude and time constraints. These will be implemented in the Phase 1 version of the EFMS after finalisation of the still ongoing prototyping and development work.

4.3.1 Horizontal Constraints

Horizontal constraints are described by the position of waypoints contained in the constraint list with additional attributes describing the size of the lateral window. At lateral

turning points the constraint data may include turn radius data. To allow path stretching to be applied as a means for time of arrival control in the TMA there are several waypoints provided with special attributes defining the path stretching area.

Horizontal constraints, i.e. position of waypoints, turn radii, are fully taken into account during the generation of the route description. In the Phase 1 version of the EFMS the route will follow great circle arcs between waypoints. The lateral size of the constraint window is therefore not used, i.e. corner cutting will not be applied.

4.3.2 CAS Constraints

Airspeed is mainly restricted by the allowed speed envelope of the respective aircraft. Besides this there are only a few additional CAS constraints steming from ATC. One is the CAS restriction to 250 kts below flight level FL100. Another one is applied at the Gate waypoint. All airspeed constraints are coped with by inclusion in the 'Phase Table', which is the major input to the prediction of airspeed and altitude profile, respectively.

4.3.3 Altitude Constraints

The application of altitude constraint windows at waypoints is defined in the 'User Requirements Document' of the EFMS [2]. Figure 6 indicates that there will exist an upper (maximum) altitude constraint and a lower (minimum) altitude constraint, which both change stepwise depending on position. Both may prevent the aircraft from following its desired altitude profile as defined by performance parameters contained in the 'Phase Table'.

The prediction starts utilising the nominal performance parameters from the initial 'Phase Table' thus resulting in the pilot's desired altitude profile. If this desired profile hits a maximum or minimum altitude constraint, then measures are taken to modify the subphases of the profile in such a way, that it conforms to the altitude constraints.

Different measures are employed depending on the phase of flight and the type of altitude constraints. The following two examples describe how altitude constraints affecting the climb at constant CAS (or Mach, respectively) are handled.

If a maximum altitude constraint (see Figure 7) affects a constant CAS climb subphase the following measures are applied:

Step 1: A constant altitude subphase is introduced, which ends when the altitude constraint is raised to a higher altitude.

Depending on the length of the level segment is decided, if it should stay in or if it should be substituted by a climb subphase comprising a smaller climb angle to avoid major thrust changes within a short time span. To achieve such a smaller climb angle there exist several solutions, such as

- increasing the climb airspeed,
- modification of Energy Sharing Index,
- reduction of thrust index.

The latter is chosen, because it allows to maintain the nominal airspeed schedule.

Step 2: The length of the level segment is checked and if it is shorter than 4 Nautical Miles (t.b.d.), the prediction of the subphase is repeated utilising a lower thrust index setting. The reduced thrust index is received by a simple formula derived from the force equations.

The thrust index TH_{IND} defined for the EFMS is as follows:

$$TH_{IND} = (THRUST - THRUST_{MIN}) / (THRUST_{MAX} - THRUST_{MIN})$$

THRUST	actual net thrust
$THRUST_{MIN}$	minimum achievable net thrust
$THRUST_{MAX}$	maximum achievable net thrust

and the reduced thrust index is then

$$TH_{IND\,RED} = TH_{IND} - W * (\gamma_{NOM} - \gamma_{REQ}) / (THRUST_{MAX} - THRUST_{MIN})$$

W	aircraft weight
γ_{NOM}	nominal flight path angle from desired climb profile
γ_{REQ}	required flight path angle

If a minimum altitude constraint (see Figure 8) affects a constant CAS/Mach climb subphase the only measure is to climb steeper. This could be achieved by either

- performing the climb subphase at a slower airspeed, or
- increasing the thrust index.

Since the thrust index is supposed to be set at 100 % during climb, corresponding to maximum continuous climb thrust, it cannot be further increased. So, the solution adopted is to reduce the climb speed accordingly.

Step 1: A first estimate of the appropriate climb CAS is derived from a consideration of total energy, which exists when the nominal climb segment hits the altitude constraint at altitude H_1. It is assumed, that the total energy available at that time could be shared in such a way, that the aircraft passes the altitude constraint at altitude H_2. This yields the improved estimated climb speed CAS_2:

$$CAS_1 = CAS\_CL1 \text{ (or } CAS\_CL2)$$

$$CAS_2 = \sqrt{\frac{\rho_{H2}}{\rho_0}} \left(\sqrt{2g(H_1 - H_2) + \left(CAS_1 \sqrt{\frac{\rho_0}{\rho_{H1}}} + V_{w,H1} \right)^2} - V_{w,H2} \right)$$

H_1	nominal altitude of flight path at constraint
H_2	required altitude at constraint
$V_{w,H}$	wind speed at altitude
g	gravity of earth
ρ_0	air density at sea level
ρ_H	air density at altitude

The prediction of the climb subphase is repeated and

yields flight path angle γ_2.

Step 2: The flight path angles received for the subphase of the desired climb profile and for the modified subphase are compared with the required flight path angle necessary to pass the altitude constraint. Linear interpolation is applied to improve the estimated climb speed.

$$CAS_3 = CAS_1 + (CAS_2 - CAS_1)*(\gamma_{REQ} - \gamma_{NOM})/(\gamma_2 - \gamma_{NOM})$$

4.3.4 Time Constraints

Time control to one or several constraint points is mainly important at the end of the route, i.e. during the final portion of the cruise flight, the descent from cruise altitude down to a Metering Fix and the final flight to the Merge Gate or the runway threshold. 4D guidance can reduce the separation distance between aircraft down to a level that approaches the minimum separation standards, since the time of arrival deviation at various constraint points can be greatly reduced. This reduction in arrival dispersion becomes a benchmark for evaluating the performance of a 4D system.

There are two basic methods of time control, that of adjusting aircraft speed [3], [4], [5] and adjusting path length [6], [7]. The choice between these two methods is driven largely by the phase of flight and the amount of time adjustment required (see Figure 9). The most economical form of time control is airspeed control in cruise flight. However, in later phases of flight there is little time remaining and thus the amount of time control possible is small. The extent to which airspeed control is a coarse or fine tuning device is, of course, a function of the airspeed range of the aircraft.

4.3.4.1 Speed Control

Time control in the cruise phase and in the descent from cruise altitude down to the Metering Fix and the Gate is achieved by altering the airspeed (CAS) within the allowed range (Minimum CAS, Maximum CAS) when predicting the trajectory between constraint points A and B (see Figure 10). This is done by applying a search strategy comprising three steps providing increasing accuracy in the estimated airspeed values and arrival times at the respective time constraint point, as shown in Figure 11. The initial prediction of the trajectory is performed applying the nominal airspeed value, i.e. CAS_CR, CAS_DE1 or CAS_DE2 respectively, contained in the 'Phase Table', thus giving the arrival time T_{CP_A} at constraint point A as well as the arrival time T_{CP1} at the time constraint point B, where the arrival time T_{CP_B} is required.

Step 1: gives a rough estimate of the appropriate airspeed value to be used for the first iteration of the profile prediction:

$$CAS_1 = CAS\_CR \text{ (or CAS\_DE1, CAS\_DE2)}$$

$$CAS_2 = CAS_1 * (T_{CP1} - T_{CP\_A})/(T_{CP\_B} - T_{CP\_A})$$

Application of CAS_2 then yields the arrival time T_{CP2}, which is used to derive a better airspeed estimate.

Step 2: Linear interpolation is applied to get the second estimate of the airspeed value to be used for the second iteration of the profile prediction:

$$CAS_3 = CAS_1 + (CAS_2 - CAS_1)*(T_{CP\_B} - T_{CP1})/(T_{CP2} - T_{CP1})$$

This results in the revised arrival time T_{CP3} at the time constraint point, which in turn can be used to improve the airspeed estimate once more, if the required time accuracy has not yet been reached.

Step 3: Quadratic interpolation is used to further improve the accuracy of the airspeed estimate. Firstly the coefficients (A, B, C) of the interpolation formula have to be defined utilising three pairs of data (CAS_1, T_{CP1}), (CAS_2, T_{CP2}), (CAS_3, T_{CP3}) and from this the speed estimate can be calculated:

$$DT = T_{CP\_B} - T_{CP\_A}$$

$$CAS_4 = A + B * DT + C * DT^2$$

This is again used to recalculate the trajectory resulting in the arrival time T_{CP4} at the time constraint.

4.3.4.2 Path Stretching

Time control in the TMA can also be achieved by altering the length of the flight path. Besides the Standard Arrival Routes (STAR) with a fixed route pattern there will exist a new type of STARs, the so-called 4D-STAR, which provides a path stretching area for the coarse and fine tuning of arrival times. The path stretching area will allow either fan type or trombone type altering of the flight path (see Figure 9 and 12).

The path stretching area is described by three waypoints with appropriate attributes:

- **Entry Waypoint WP_{EN}**
  The Entry Waypoint is part of the 4D-STAR and defines the entry into the path stretching area.

- **Exit Waypoint WP_{EX}**
  The Exit Waypoint is part of the 4D-STAR. It defines together with the Intercept Waypoint WP_{CP} the direction in which the aircraft will leave the path stretching area.

- **Intercept Waypoint WP_{CP}**
  The Intercept Waypoint describes where the aircraft has to turn towards the path stretching Exit Waypoint WP_{EX}. Its nominal position $WP_{CP\text{-}NOM}$ is defined by the 4D-STAR.

To provide for time of arrival control the position of the End of Turn point EOT_{CP} is shifted appropriately along the straight line segment between WP_{CP} and WP_{EX} within a specified range around its nominal position.

In the Phase 1 version of the EFMS path stretching will only be applied in the course of trajectory regeneration at the Metering Fix or when the aircraft reaches one of the two

Update Waypoints (WP_{UPD1} or WP_{UPD2}).

- **Update Waypoint WP_{UPD1}** is identical to the position of the Start of Turn at the Entry Waypoint WP_{EN}.

- **Update Waypoint WP_{UPD2}** is situated about 40 seconds before reaching the Start of Turn SOT_{CP} at Intercept Waypoint WP_{CP}.

Path regeneration is performed iteratively applying a search strategy comprising three steps, as shown in Figure 13. Curves A and B in Figure 13 describe the dependancy between arrival time T at the Gate and distance D between Gate and End of Turn point EOT_{CP}. Curve A corresponds to the previously predicted trajectory yielding arrival time T_1 at the Gate, providing an accuracy, which has been regarded sufficient, because of the wider time horizon of that previous prediction. Since the aircraft follows the 4D-trajectory only within certain tolerances, curve A moves sidewards depending on observed error in arrival time. To compensate an arrival time error ΔT, as shown in the figure, distance D has to be modified stepwise and this search would be along curve B.

The search starts from the nominal flight path in the path stretching area, which is understood to be the path along the published 4D-STAR without any path modifications. From this the nominal distance D_1 between the Exit Waypoint WP_{EX} and the End of Turn point EOT_{CP} as well as the predicted arrival time at the Gate T_1 are taken. The following steps are performed to predict the path which conforms to the required arrival time at the Gate T_{GT}.

Step 1: Depending on the observed arrival time error ΔT at the update point the position of the End of Turn point EOT_{CP} is shifted along the straight segment between WP_{CP} and WP_{EX}.

$$D_2 = D_1 + \Delta T * V_{GS} - \Delta D/2$$

V_{GS} ground speed on the straight segment between WP_{CP} and WP_{EX}

ΔD interpolation interval, e.g. 200 meters

The flight path between the update point and the Gate is then recalculated to give the arrival time T_2 at the Gate.

Step 2: The position of the final End of Turn point EOT_{CP} is shifted by the interpolation interval ΔD:

$$D_3 = D_2 + \Delta D$$

Flight path as well as arrival time T_3 at the Gate are recalculated.

Step 3: The estimated position of the End of Turn point EOT_{CP} for the time accurate path is then produced by linear interpolation.

$$D_4 = D_2 + \Delta D*(T_{GT} - T_2)/(T_3 - T_2)$$

The time accurate path to the Gate can then be predicted resulting in arrival time T_4.

4.4 Planning Steps for a Complete 4D-Trajectory

The prediction of a complete 4D-trajectory, which conforms to several airspeed, altitude and time constraints requires certain subphases of the vertical profile to be calculated iteratively. This iterative process consists of an inner loop, which aims at meeting horizontal, airspeed and altitude constraints and an outer loop to cope with time constraints. The example shown in Figure 14 comprises 16 planning steps. Arrow heads indicate forward or backward prediction.

The generation of the initial trajectory taking horizontal, airspeed and altitude constraints into account consists of planning steps 1, 2, 3, and 4.

The time constraint at the Cruise Fix CF is dealt with by iteratively modifying the cruise speed CAS_CR, as described in chapter 4.3.4.1, comprising planning steps 5, 6 and 7.

The time constraints at the Gate and at the Metering Fix are dealt with by modification of descent speed CAS_DE1 and CAS_DE2, respectively. The cruise speed applicable for the segment between Cruise Fix CF and Cruise Deceleration Point CDP may also be modified, if regarded necessary. This part of the planning comprises planning steps 8, 9 and 10, which are then iteratively repeated (steps 11, 12, 13, and steps 14, 15, 16).

Regeneration of the trajectory in flight will be initiated depending on certain events or when the aircraft arrives at predefined update points. The scheme of trajectory prediction will still be as described above, but it will start at present position of the aircraft and the prediction of subphases already passed will be left out, of course. The criteria, when e.g. to stop the iterative search for an appropriate airspeed for time of arrival control will depend on flight progress, i.e. time horizon of planning. The required accuracy of the predicted 4D-trajectory as regards arrival time at a certain time constraint point will be increased, when approaching that respective time constraint point, thus to provide a symmetrical margin for likely deviations from the trajectory.

5. Conclusions

This paper describes briefly the functionality of an experimental Flight Management System (EFMS) disctinct for research into future Air Traffic Management concepts. The development of this EFMS as such, is a major research and development task. Thus it was decided to split up the development of the EFMS into several phases. The development of a Phase 1 version of the EFMS with a limited functionality is underway. This Phase 1 version will already be applicable for initial investigations including flight trials with experimental aircraft. The strategy for the prediction of 4D-trajectories will be as described in this paper. A later version of the EFMS will have a more complete functionality and may comprise more advanced planning strategies to be developed in the course of the planned research activities with the Phase 1 version of the EFMS.

6. List of References

[1] CENA, DLR, NLR, RAE, EUROCONTROL
'A Conceptual Description of an Experimental FMS for Air Traffic Management Research', EUROCONTROL, 1988.

[2] CENA, DLR, NLR, RAE, EUROCONTROL
'User Requirements Document of an Experimental FMS for Air Traffic Management Research', Version 4.1, EUROCONTROL DOC 912008, 1991.

[3] Knox, C.E., Cannon, D.G.
'Development and Test Results of a Flight Management Algorithm for Fuel Conservative Descents in a Time Based Metered Traffic Environment', NASA TP 17171, 1980.

[4] Erzberger, H., Tobias, L.
'A Time-Based Concept for Terminal Area Traffic Management', AGARD, CP 410, 1986.

[5] Lee, H.P., Leffler, M.F.,
'Development of the L-1011 Four-Dimensional Flight Management System', NASA CR 3700, Febr. 1984.

[6] Lechner, W.
'Algorithms for the Automatic 4-Dimensional Guidance of Aircraft Taking into Account the Current Wind Situation', ESA-TT-908, October 1985.

[7] Adam, V.
'Onboard Planning and Control of 4D-Trajectories in the TMA', DLR-Mitt. 89-23, 1989.

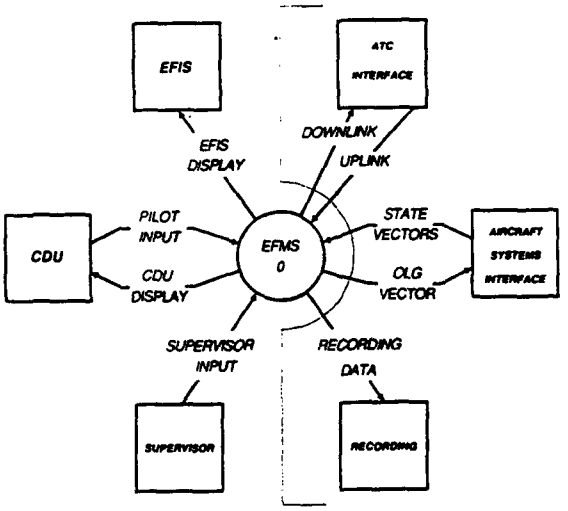


Figure 1. Interfaces of the EFMS

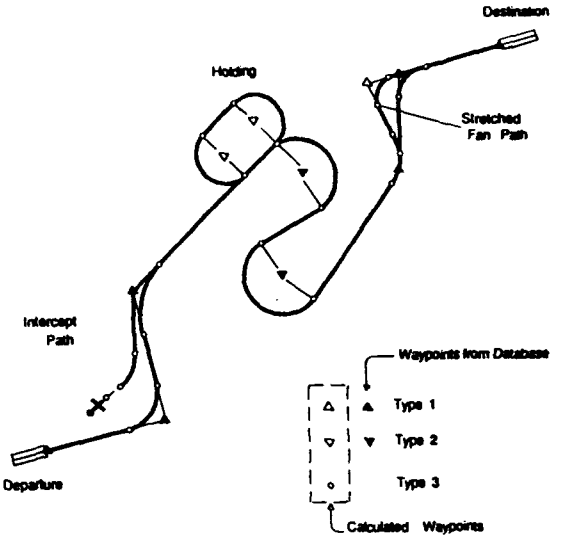


Figure 3. Generic Route

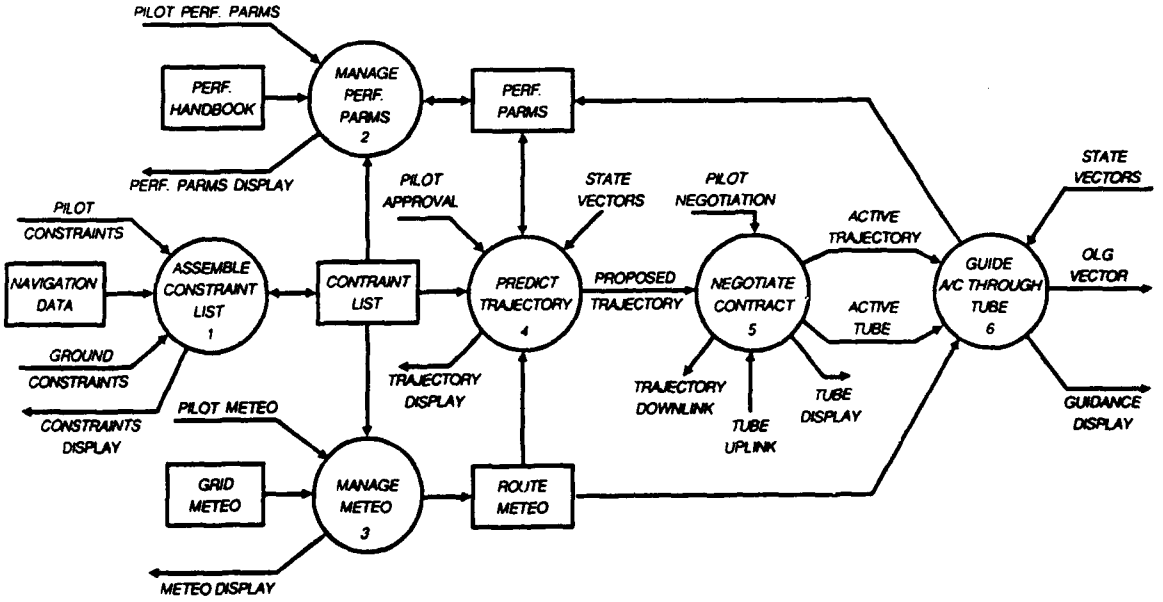


Figure 2. Internal Structure of the EFMS (Level 0)

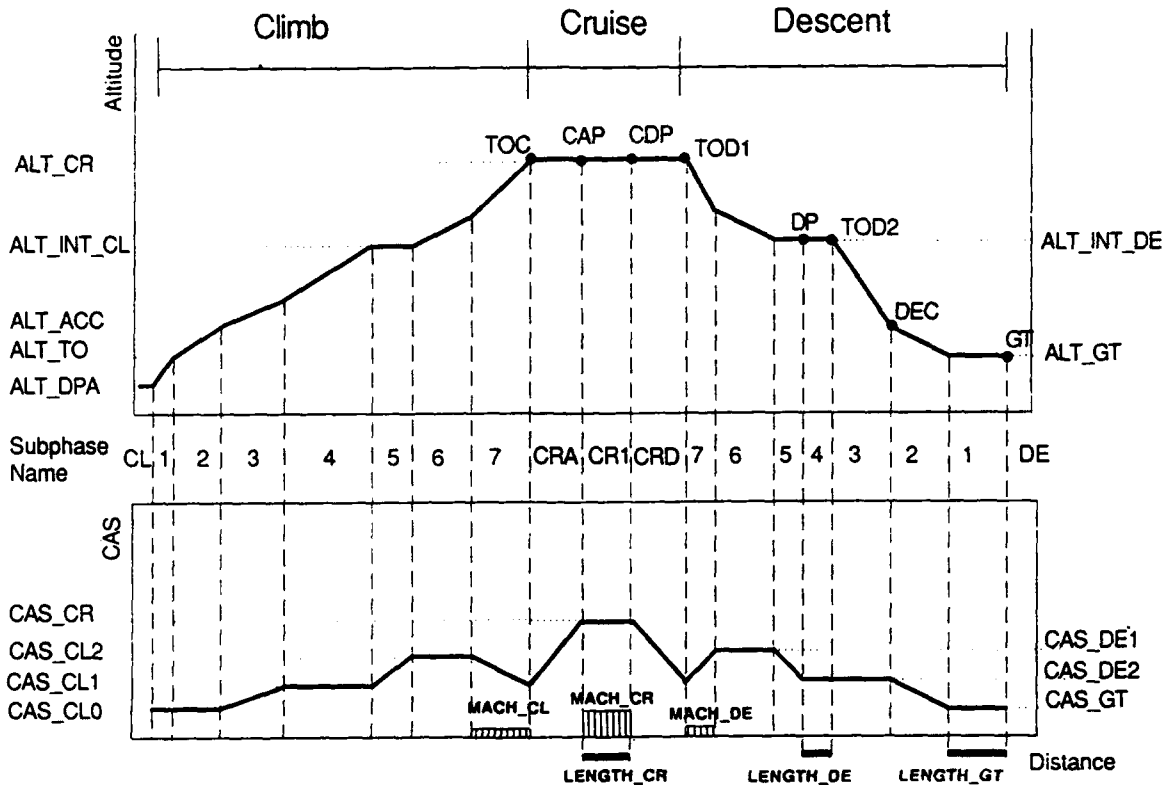


Figure 4. Generic Altitude Profile

SUB PHASE NAME	SUBPHASE_CONDITION_INDEX				SUBPHASE_TARGETS				SUBPHASE_EXIT_CONDITIONS			
	CONSTANT _PARMS _INDEX	THRUST _MODE	THRUST _INDEX	CONFIG _INDEX	ALT	CAS	MACH	ESI	ALT	CAS	MACH	LENGTH
CL 1	CAS	TO	1.0	TO	---	CAS_CL0	---	---	ALT_TO	---	---	---
2	CAS	MC	1.0	CLEAN	---	CAS_CL0	---	---	ALT_ACC	---	---	---
3	ESI	MC	1.0	CLEAN	---	---	---	ESI_CL	ALT_INT_CL	CAS_CL1	---	---
4	CAS	MC	1.0	CLEAN	---	CAS_CL1	---	---	ALT_INT_CL	---	---	---
5	ALT	MC	1.0	CLEAN	ALT_INT_CL	---	---	---	---	CAS_CL2	---	---
6	CAS	MC	1.0	CLEAN	---	CAS_CL2	---	---	ALT_CR	---	MACH_CL	---
7	MACH	MC	1.0	CLEAN	---	---	MACH_CL	---	ALT_CR	---	---	---
CRA	ALT	MC	1.0	CLEAN	ALT_CR	---	---	---	---	CAS_CR	---	---
CR 1	ALT,CAS	AR	---	CLEAN	ALT_CR	CAS_CR	---	---	---	---	---	LENGTH_CR
CRD	ALT	IDLE	0.0	CLEAN	ALT_CR	---	---	---	---	---	MACH_DE	---
DE 7	MACH	IDLE	0.0	CLEAN	---	---	MACH_DE	---	ALT_CR	---	---	---
6	CAS	IDLE	0.0	CLEAN	---	CAS_DE1	---	---	ALT_CR	---	MACH_DE	---
5	ALT	IDLE	0.0	CLEAN	ALT_INT_DE	---	---	---	---	CAS_DE1	---	---
4	ALT,CAS	AR	---	CLEAN	ALT_INT_DE	CAS_DE2	---	---	---	---	---	LENGTH_DE
3	CAS	IDLE	0.0	CLEAN	---	CAS_DE2	---	---	ALT_INT_DE	---	---	---
2	ESI	IDLE	0.0	CLEAN	---	---	---	ESI_DE	ALT_INT_DE	CAS_DE2	---	---
1	ALT,CAS	AR	---	APP	ALT_GT	CAS_GT	---	---	---	---	---	LENGTH_GT

Figure 5. Initial Phase Table

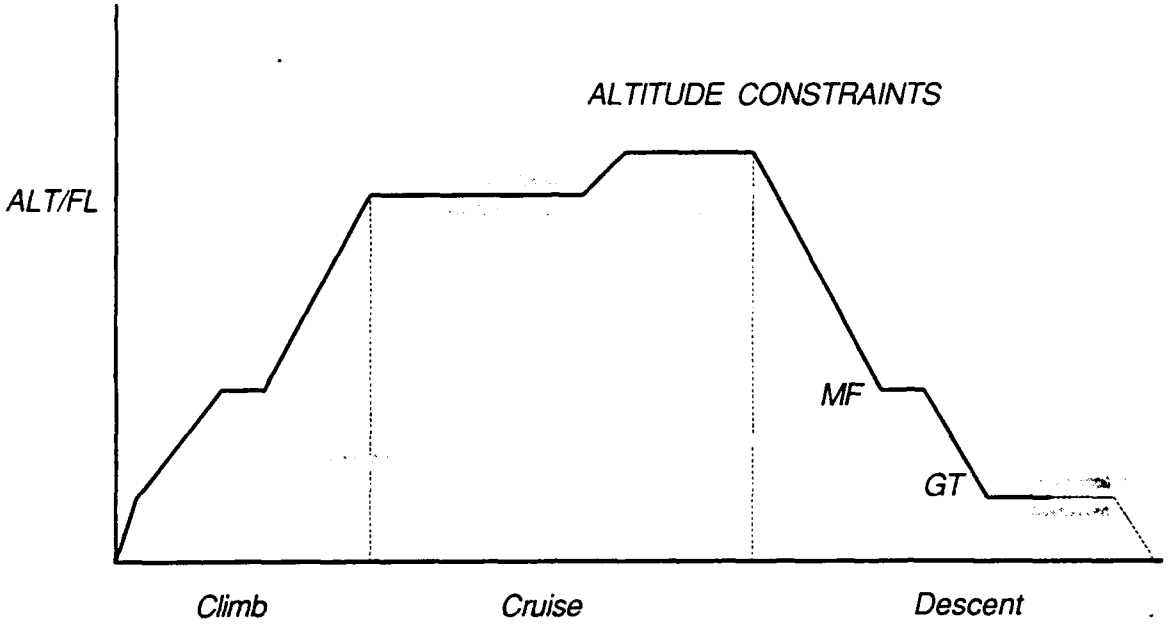


Figure 6. Altitude Profile and Altitude Constraints

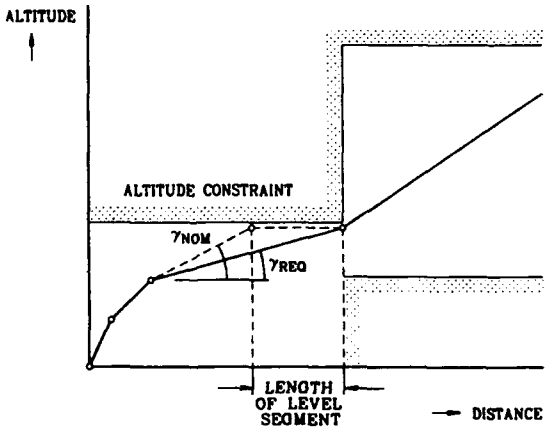


Figure 7. Strategy to Overcome a Maximum Altitude Constraint

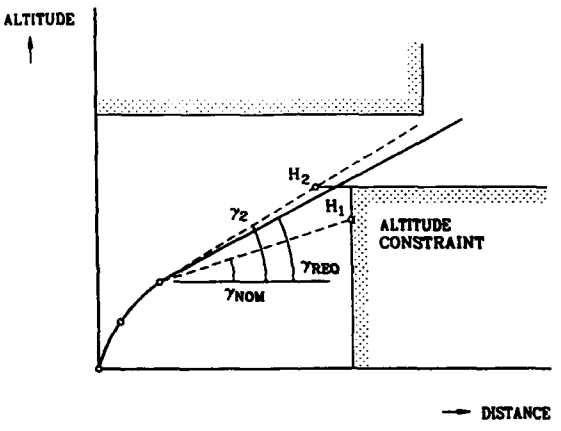


Figure 8. Strategy to Overcome a Minimum Altitude Constraint

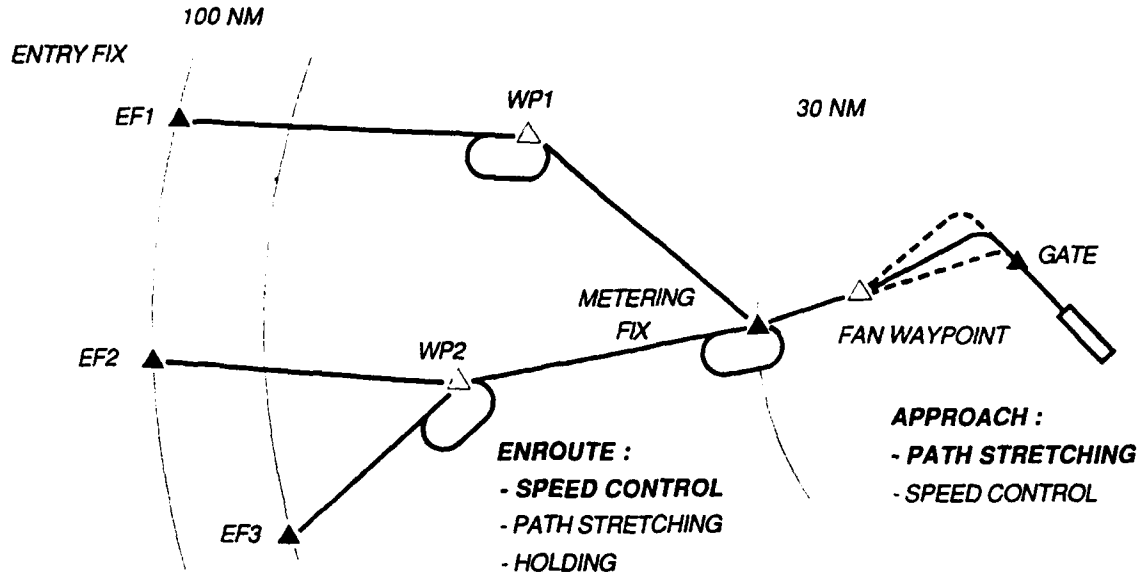


Figure 9. Measures for Time of Arrival Control

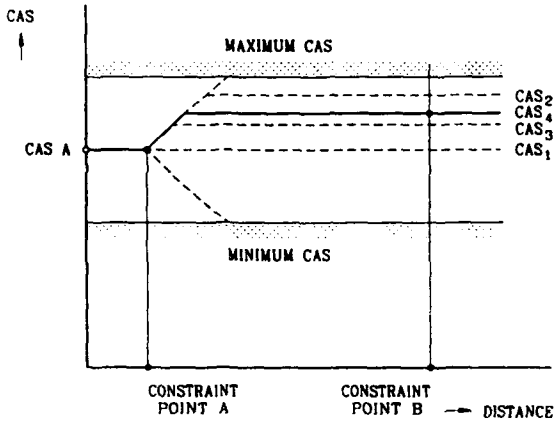


Figure 10. Variation of CAS for Control of Arrival Time

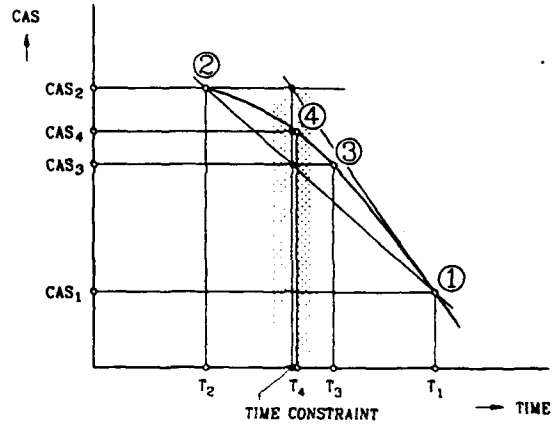


Figure 11. Strategy to Define CAS for Control of Arrival Time

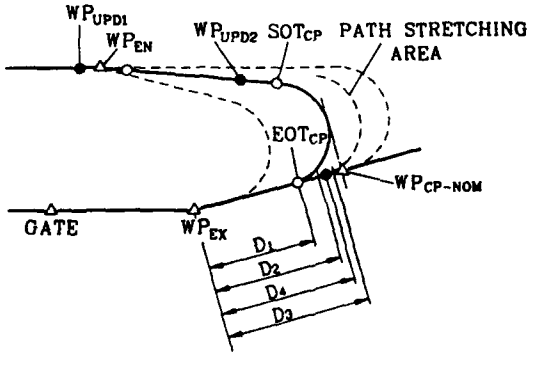


Figure 12. Trombone Type Path Stretching

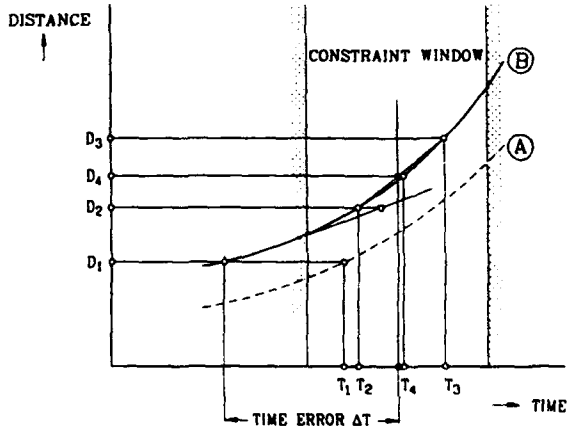


Figure 13. Strategy to Define the Stretched Path for Control of Arrival Time

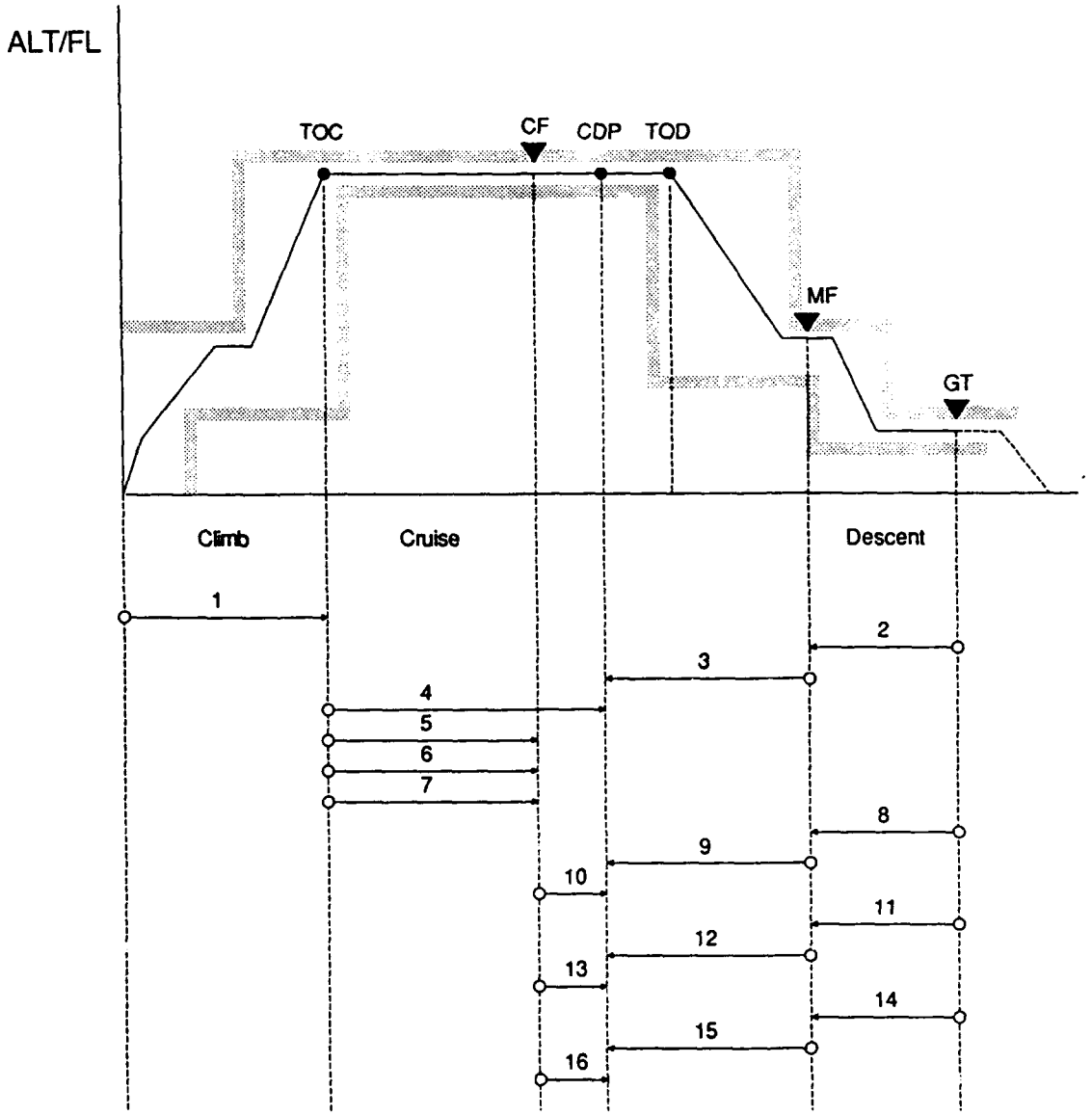


Figure 14. Strategy for the Prediction of a Complete 4D-Trajectory which Conforms to Altitude Constraints and Time Constraints

Trajectory Optimization for Hypersonic Aircraft Guidance

R.L. Schultz, M.J. Hoffman, A.M. Case, and S.L. Sheikh
Honeywell Systems and Research Center
3660 Technology Drive
Minneapolis, Minnesota USA 55418

SUMMARY

A computationally efficient, real-time trajectory optimization and guidance approach for hypersonic aircraft is described. The optimization approach is based on Euler-Lagrange energy state approximations. A three-dimensional, spherical-earth, aircraft-motion model with constraints on temperature, dynamic pressure, acceleration, and angle of attack is employed. Climb-to-orbit, return-from-orbit, flight-to-designated-landing-site, unpowered-abort, and powered-abort flight conditions are considered. Different performance criterion are used for different problems.

Solution methods of varying computational complexity and performance capability are developed. An exact solution to the optimization problem using iteration on adjoint variables is developed. This method is the most exact but is not suitable for on-board processing. However, it serves as the basis of performance comparison for the approximate methods. Approximate solution methods suitable for onboard guidance are developed for the climb-to-orbit problem and the flight-to-a-landing site problem.

A computationally efficient method for generating footprints is described. Footprints are used to determine when to start final descent and to identify candidate landing sites under an air-breathing-engine or rocket-engine abort or other emergency conditions. A hypersonic-vehicle guidance, navigation, and control configuration of the complete optimal guidance scheme is described. Sensitivity analysis results are included for climb-to-orbit trajectories. The probability of achieving orbit using the optimal guidance scheme and a stored nominal approach are compared.

This research was sponsored by the Air Force Wright Laboratory.

LIST OF SYMBOLS

a	Speed of sound
a_{nc}	Normal (altitude) acceleration command
a_{yc}	Lateral acceleration command
A_c	Cowl area of air-breathing engine
C_a	Capture area of air-breathing engine
C_{D0}	Parasite drag coefficient
C_L	Coefficient of lift
D	Drag
E	Specific energy
F_z	Normal forces
$F_{z\alpha}$	Partial derivative of the normal forces with respect to angle of attack
g	Gravity
$\tilde{g}$	gravity minus orbital relief acceleration $(g - V^2/R)$
G	Performance criteria
h	Altitude
I_{sp_a}	Air-breather specific impulse
I_{sp_r}	Rocket specific impulse
J	Performance integral
k	Induced drag coefficient
L	Lift
m_h	Mass of hydrogen
m_o	Mass of oxygen
M	Mach number
M_a	Aerodynamic moments
M_y	Sum of pitching moments
Q	Dynamic pressure
R	Radius from the center of the earth ($R = R_0 + h$)
R_0	Radius of the earth
S	Wing reference area
T	Thrust
t	Time
u	Control vector
u_h	Mass flow rate of hydrogen
u_o	Mass flow rate of oxygen
V	Velocity
x	State vector
z	Roll control variable ($z = \tan\sigma$)
θ	Longitude
ϕ	Latitude or cross-range position
α	Angle of attack
δ	Elevator deflection
σ	Roll angle
γ	Flight path angle
λ	Adjoint
ρ	Density
π	Throttle
ψ	Heading angle (East = 0 deg)

Subscripts

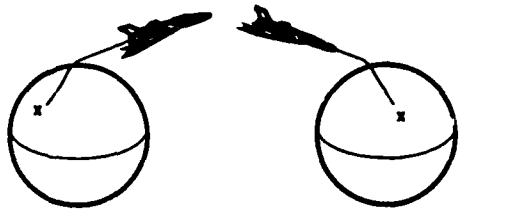
a	Air-breathing engine
f	Final
h	Hydrogen
o	Oxygen
r	Rocket engine
0	Initial

INTRODUCTION

A key element of optimizing performance and operability of hypersonic aircraft is the ability to generate onboard flight trajectories that optimize energy management in response to

mission phase demands and contingencies. The ability to compute these trajectories on board is a step toward providing the autonomous capability required for hypersonic vehicle operations. Past space vehicle programs, such as Space Shuttle, lacked this autonomy and consequently required complex ground support services. A truly autonomous aerospace vehicle would greatly reduce this associated cost by providing the capability to accommodate takeoff delays, changing mission conditions, low-fuel conditions, engine failures, and other emergencies.

The objectives of the hypersonic aircraft guidance solution are to fly from a given latitude, longitude, altitude, and velocity to a specified final condition while minimizing some performance objective, and to continuously compute, in-flight, the optimal trajectories between the changing current vehicle state conditions and the specified final state. The optimization problem, then, is one of managing kinetic, potential, thermal, and chemical energy. For onboard hypersonic guidance there are two vehicle optimization problems to be considered: climb-to-orbit and flight-to-landing-site. Figure 1 depicts both problems.



Climb to Orbit

- Final conditions
  - Energy: E_f
  - Position: None
  - Fuels: $m_o \geq 0$ and $m_h \geq 0$
- Planar problem
- Manage fuels to reach orbit with uncertainties

Flight to Landing Site

- Final conditions
  - Energy: E_f
  - Position: θ_f and ϕ_f
  - Fuels: $m_h \geq 0$ and $m_o \geq 0$
- Three-dimensional problem
- Predict if desired landing site can be reached with current fuel conditions
- Guide vehicle to selected landing site

Figure 1. Hypersonic Aircraft Guidance Optimization

The steps in the development approach are to:

1. Obtain optimal solutions to establish the upper bound of attainable performance and establish the characteristics of the optimal trajectories.
2. Assess the feasibility of using optimization for onboard guidance.
3. Define approximations to optimal strategies and assess the computational requirements.
4. Select the best approach based on trade between performance and processor requirements.

The general Euler-Lagrange optimization method is described below. Then, solutions for the specific problems of climb-to-orbit and flight-to-landing-site are presented. Finally, a complete vehicle in-flight guidance system is described. The complete onboard system combines both solution approaches and additional features such as real-time footprints.

GENERAL EULER-LAGRANGE OPTIMIZATION METHOD

There are several optimization methods available for solving optimal guidance problems. Table A-1 in Appendix A summarizes some of these methods. The solution approach used is in this study is the Euler-Lagrange method. Further details on the problem definition and solution can be found in reference 1.

Solution to the Optimization Problem

The general statement of the optimization problem and its solution are stated below.

Minimize the performance integral, J,

$$J = \int_0^{t_f} G(x,u,t)dt$$

subject to these constraints:

$$\dot{x} = f(x,u,t) \quad ; \quad x(t_0) = x_0$$

$$C(x,u,t) < 0$$

The solution to this optimization problem is the following set of Euler-Lagrange equations.

The controls are found by minimizing the Hamiltonian:

$$\min_u (H)_x$$

The Hamiltonian is given by

$$H = G + \lambda^T f$$

The adjoint variables λ are given by

$$\dot{\lambda} = -\frac{\partial H}{\partial x}$$

The boundary conditions on λ and x are given by the transversality conditions

$$\left[Hdt - \lambda^T dx \right]_{t_0}^{t_f} = 0$$

When H is independent of time, it must also satisfy the first integral condition $\dot{H} = 0$.

The hypersonic aircraft problem

The objectives of the hypersonic aircraft guidance problem are to fly from a given latitude, longitude, altitude, and velocity to a specified final condition while minimizing some performance objective, and to continuously compute, in-flight, the optimal trajectories between the changing current vehicle state conditions and the specified final state.

Optimization criteria

Within the general Euler-Lagrange problem, specific problems for hypersonic vehicles may have different performance criteria and/or different specified end conditions, as shown in Table 1.

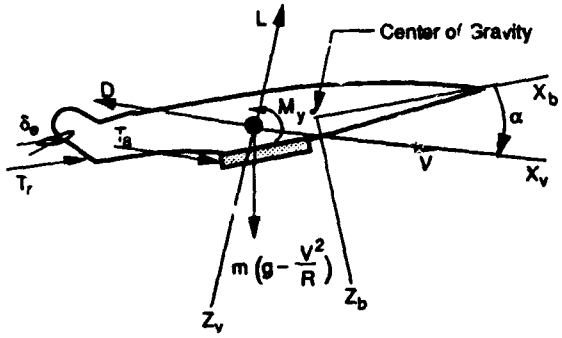
Vehicle models

The vehicle model is defined by the equations of motion, engine model, aerodynamic model and constraints. The vehicle configuration is defined in Figure 2.

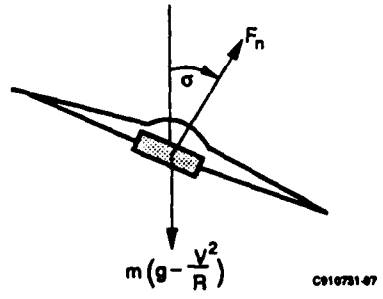
Table 1. Euler-Lagrange Optimization Criteria

Performance Objective	Performance Integral	Terminal Condition
Minimum fuel climb-to-orbit	$J = \int_0^{t_f} (\dot{m}_h + \dot{m}_o)\, dt$	E_f final energy
Minimum fuel climb-to-orbit with specified final value of H_2 or O_2	$J = \int_0^{t_f} (\dot{m}_h + \dot{m}_o)\, dt$	E_f final energy m_h final mass of H_2 or m_o final mass of O_2
Minimum fuel to a specified downrange and crossrange	$J = \int_0^{t_f} (\dot{m}_h + \dot{m}_o)\, dt$	E_f final energy θ_f final longitude ϕ_f final latitude
Minimum sum of distance traveled, weighted drag squared and roll angle squared	$J = \int_0^{t_f} \left[C_a V + w_d \left(\frac{D}{m}\right)^2 + k_2 z^2\right] dt$	E_f final energy θ_f final longitude ϕ_f final latitude
Footprint Calculation: maximize crossrange for a given downrange	$J = \int_0^{t_f} \dot{\phi}\, dt$	E_f final energy θ_f final downrange

C91-0791-1A



a. Side View



b. Front View

Figure 2. Vehicle Configuration

The equations of motion for a spherical, rotating earth are as follows:

$$\dot{V} = \frac{T_x - D}{m} - g \sin\gamma + \omega^2 r \cos\phi\, (\sin\gamma \cos\phi - \cos\gamma \sin\phi \sin\psi)$$

$$\dot{\gamma} = \frac{(L - T_z)\cos\sigma}{mV} - \left(g - \frac{V^2}{R}\right)\frac{\cos\gamma}{V} + 2\omega \cos\psi \cos\phi + \frac{\omega^2 r \cos\phi}{V}(\cos\gamma \cos\phi + \sin\gamma \sin\psi \sin\phi)$$

$$\dot{\psi} = \frac{(L - T_z)\sin\sigma}{m V \cos\gamma} - \frac{V}{R}\cos\gamma \cos\psi \tan\phi + 2\omega(\tan\gamma \sin\psi \cos\phi - \sin\phi) - \frac{\omega^2 r \cos\psi \sin\phi \cos\phi}{V \cos\gamma}$$

$$\dot{h} = V \sin\gamma$$

$$\dot{\theta} = \frac{V \cos\gamma \cos\psi}{R \cos\phi} \quad ; \quad R = R_0 + h$$

$$\dot{\phi} = \frac{V \cos\gamma \sin\psi}{R}$$

$$\dot{m}_h = u_{h_a} + u_{h_r}$$

$$\dot{m}_o = u_{o_r}$$

The eight states are: velocity (V), flight path angle (γ), heading (ψ), altitude (h), longitude (θ), latitude (ϕ), mass of hydrogen (m_h), and mass of oxygen (m_o).

The five control variables are: angle of attack (α), elevator deflection (δ), roll angle (σ), air-breathing engine throttle (π_a), and rocket engine throttle (π_r).

The general form of the force and moment equations are:

$$\begin{aligned} L &= QSC_L(\alpha,\delta,M) \\ D &= QSC_D(\alpha,\delta,M) \\ T_a &= T(\pi_a,\alpha,M,\rho,a) \\ T_r &= T_r(\pi_r) \\ M_a &= QS\bar{c}C_m(\alpha,\delta,M) \\ M_{T_a} &= M_{T_a}(\pi_a,\alpha,\delta,M,\rho,a) \\ M_{T_r} &= M_{T_r}(\pi_r) \end{aligned}$$

The air-breathing and rocket thrusts in the body axes are expressed as follows:

$$T_a = I_{sp_a} K_t \pi_a g \rho M a C_a A_c$$

$$T_r = T_{r_{max}} \pi_r$$

In the velocity axes, the total thrust components are:

$$T_x = T_{ax} + T_{rx}$$

$$T_z = k_c T_{ax} + T_{rz} \qquad \text{where } k_c \cong 0.3$$

The hydrogen and oxygen fuel flow rates are:

$$u_h = \frac{T_a}{I_{sp_a} g} + \frac{T_r}{9 I_{sp_r} g}, \; u_o = \frac{8 T_r}{9 I_{sp_r} g}$$

The aerodynamic drag (D) and lift (L) functions are expressed as

$$D = QS[C_{D0} + K\alpha^2 + K_\delta \delta^2]$$

or

$$D = QS[C_{D0} + \bar{K} C_L^2]$$

$$L = QSC_L(M,\alpha,\delta)$$

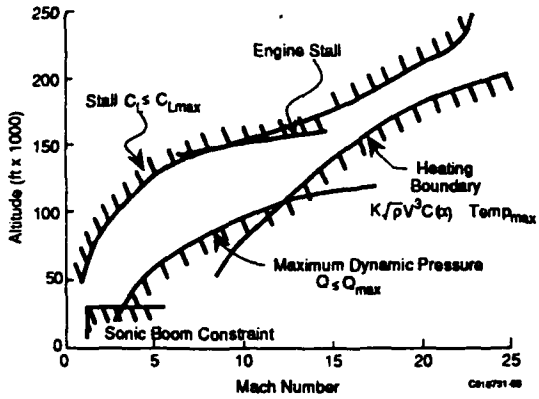
The total pitching moment is of the form

$$M_y = QS\bar{c}C_m + T_{ax} z_a + T_{az} x_a + T_{rx} z_r + T_{rz} x_a$$

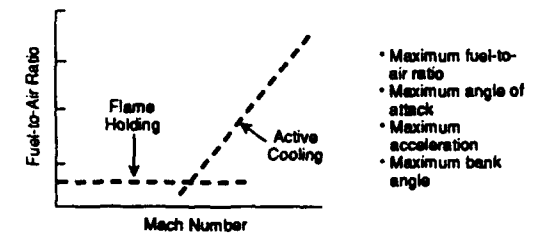
Constraints

The constraints on the trajectory variables may be of the equality or inequality type. The system must operate in such a way that the resulting trajectory does not violate the constraints. Some typical hypersonic vehicle constraints are shown in Figure 3. The plot (a) shows some of the constraints

on an altitude-velocity grid. Plot (3b) shows the throttle constraints. The flame-holding constraint prevents the fuel-to-air ratio from becoming so small that the engine ceases operating. The other throttle constraint is an active-cooling constraint. Fuel flow must be retained if active cooling is employed.



a. Constraints Related to Altitude and Velocity



b. Minimum Throttle Constraints

c. Additional Constraints

Figure 3. Typical Hypersonic Vehicle Constraints

The dynamic pressure is limited to be less than a maximum value and greater than a minimum value as follows:

$$Q_{min} \le 0.5\rho V^2 \le Q_{max}$$

The temperature due to aerodynamic heating is constrained to be below a maximum value as follows:

$$Temp \le Temp_{max}$$

where

$$k_2(Temp)^4 = k_1\sqrt{\rho}\, V^{3.05}\, C(\alpha)$$

Energy-State Approximations

To reduce the difficulties of solving the two point boundary value problem resulting from the full Euler-Lagrange solution equations, an approach based on simplifying approximations is considered. In this approach, assumptions and simplifications are made to reduce the number and complexity of the adjoint variables.

In the energy-state approximations, it is assumed that vertical motion has higher frequency dynamics than horizontal motion. This assumption is similar to the assumption that the attitude dynamics are at a higher frequency than the positional dynamics. Ultimately, both assumptions imply that *an attitude controller and a vertical motion controller* are supplied where the control frequencies of the successively higher dynamics are separated by a factor of 10 or more. The γ and h differential equations are eliminated and replaced by their equilibrium conditions. The energy-state equations are:

$$\dot{E} = (T - D)\frac{V}{m} \qquad ; \quad E = \frac{V^2}{2} + gh$$

$$\dot{\psi} = \frac{\tilde{g}}{V}\tan\sigma - \frac{V}{R_0}\cos\psi\tan\phi \qquad ; \quad \tilde{g} = g - \frac{V^2}{R_0}$$

$$\dot{\theta} = \frac{V\cos\psi}{R_0\cos\phi}$$

$$\dot{\phi} = \frac{V\sin\psi}{R_0} \qquad ; \quad R = R_0 + h$$

$$\dot{m}_h = u_{h_s} + u_{h_r}$$

$$\dot{m}_O = u_{O_T}$$

The energy-state assumptions result in vertical force and pitching moment equality constraints:

$$F_z = L - T_{az} - T_{rz} - m\tilde{g} = 0$$

$$M_y = QS\bar{c}C_m + T_{ax}z_a + T_{az}x_a + T_{rx}z_r + T_{rz}x_a = 0$$

Using vertical equilibrium, the maximum acceleration and stall inequality constraints become:

$$\frac{\frac{m\tilde{g}}{\cos\sigma} + T_z}{0.5\rho V^2 S} \le C_{L_{max}} \quad ; \quad \frac{m\tilde{g}}{\cos\sigma} \le a_{n_{max}}$$

With the energy-state equations of motion, the Hamiltonian reduces to:

$$H = G + \lambda_\theta\frac{V\cos\psi}{R_0\cos\phi} + \lambda_\phi\frac{V\sin\psi}{R_0} + \lambda_E(T_x - D)\frac{V}{m}$$
$$+ \lambda_\psi\left[\frac{\tilde{g}}{V}\tan\sigma - \frac{V^2}{R_0}\cos\psi\tan\phi\right] + \lambda_{m_h}\dot{m}_h + \lambda_{m_o}\dot{m}_o$$

The adjoint equations λ_h and λ_γ have been eliminated. The remaining adjoint equations are:

$$\dot{\lambda}_E = -\frac{\partial H}{\partial E}$$

$$\dot{\lambda}_\theta = 0$$

$$\dot{\lambda}_\phi = \lambda_\theta\frac{V\cos\psi\sin\phi}{R_0\cos^2\phi} - \lambda_\psi\frac{V}{R_0}\cos\psi\sec^2\phi$$

$$\dot{\lambda}_\psi = -\lambda_\phi\frac{V}{R_0}\cos\psi + \lambda_\theta\frac{V\sin\psi}{R_0\cos\phi} - \lambda_\psi\frac{V}{R_0}\sin\psi\tan\phi$$

$$\dot{\lambda}_{m_h} = -\frac{\partial H}{\partial m_h}$$

$$\dot{\lambda}_{m_o} = -\frac{\partial H}{\partial m_o}$$

Further Simplifications

By applying the energy-state assumption, the number of states has been reduced and two adjoint equations have been eliminated. The Euler-Lagrange energy-state equations can be further simplified as follows.

Analytical solution to adjoint equation

If the performance criterion, G, does not explicitly include the states θ, ϕ or ψ, it is possible to solve the λ_θ, λ_ϕ, and λ_ψ differential adjoint equations in closed form. The solution is derived in Vinh (Reference 2). From Vinh, the solution takes the form:

$$\lambda_\theta = C_1$$

$$\lambda_\phi = C_2\sin\theta - C_3\cos\theta$$

$$\lambda_\psi = C_1\sin\phi + (C_2\cos\theta + C_3\sin\theta)\cos\phi$$

Reformulation of adjoint constants of integration

The adjoint constants are expressed in terms of three other constants that have physical meaning. From Vinh, the adjoint constants form a vector that is perpendicular to the final position and velocity vector. The final condition on λ_ψ (λ_ψ = 0) is used to eliminate one parameter. The constants in terms of the new parameters are:

$$C_1 = \cos\phi_f \cos\psi_f C_0$$

$$C_2 = (\sin\theta_f \sin\psi_f - \cos\theta_f \sin\phi_f \cos\psi_f) C_0$$

$$C_3 = (-\cos\theta_f \sin\psi_f - \sin\theta_f \sin\phi_f \cos\psi_f) C_0$$

ψ_f is the final heading angle and C_0 is related to the cruise minimum-fuel flow rate.

Elimination of the adjoint constant λ_E

The adjoint constant λ_E is eliminated using the minimum principle and the first integral as follows:

$$H(u^*) \le H(u^* + \Delta u) \; ; \; H = 0$$

where u^* is the optimal control and $u^* + \Delta u$ is the non-optimal control. The derivation, which is presented in Reference 3, results in two possible operations on a new Hamiltonian H_a:

$$\min_u [H_a(u)] \; ; \text{ if } \dot{E} > 0$$

$$\max_u [H_a(u)] \; ; \text{ if } \dot{E} < 0$$

Selection of extremum operation

The elimination of the adjoint variable λ_E results in two possible extremum operations. The condition for switching between extremum operations can be determined using the continuity of λ_E and $\lambda_E = -\min(H_a)$ or $\lambda_E = -\max(H_a)$. The switching condition, which is derived in Reference 3, is

$$\max (H_a)_{h<0} = \min (H_a)_{h>0}$$

The starting operation is still unknown at this point.

Finding extrema of the Hamiltonian

The optimal controls are found by determining the extrema of the Hamiltonian with respect to the controls:

$$\underset{\pi_a \, \pi_r \, h \, \sigma}{\text{Extremum}} \left(H_a\right)_{\substack{F_z=0 \\ M_y=0}} ; \quad \text{Extremum} = \begin{cases} \text{maximize if } \dot{E}<0 \\ \text{minimize if } \dot{E}\ge 0 \end{cases}$$

Summary of the General Energy-State Solution Method

In summary, the steps in the solution of the general hypersonic guidance problem are the following.

1. Choose starting operation (i.e., maximize or minimize H_a).
2. Pick initial values of adjoint constants (ψ_f, C_o) and fuel adjoints λ_{m_o}, λ_{m_h}.
3. Compute minimum of Hamiltonian with respect to controls (π_a,π_r,h,σ).
4. Compute angle of attack and elevator from vertical and moment equilibrium.
5. Increment equations of motion to next time point.
6. If necessary, switch optimizing operations when they are equal.
7. Stop integration at final energy E_f.
8. Compute misses on final conditions (i.e., specified position, fuel masses, adjoints).
9. Update adjoints using final misses.
10. Repeat (from 3) until misses are small.

In the next two sections, the solutions to specific hypersonic optimization problems are discussed. We will see that many of the above listed steps are eliminated for certain problems, such as minimum fuel climb-to-orbit. Also, many of the above steps can be accurately approximated using state feedback and switching logic.

SOLUTION FOR CLIMB TO ORBIT

The solution for minimum-fuel climb to orbit is developed. The goal is to reach a final orbital energy using the minimum amount of fuel without violating the constraints. First, the exact optimal solution is derived, and then approximate solution methods are described. Finally, recommendations for onboard guidance are given and simulation results are presented.

Exact Climb-to-Orbit Energy State Solutions

The performance integral for the minimum-fuel climb-to-orbit problem is

$$J = \int_0^{t_f} (u_h + u_o) \, dt$$

where

$$u_h = \dot{m}_h \; ; \; u_o = \dot{m}_o$$

The terminal conditions define the exact problem to be solved. If neither final fuel mass is specified, the final energy, E_f, is the only terminal condition. If, however, the final mass of one of the two fuels is specified, there are two terminal conditions: E_f and either $m_h(t_f)$ or $m_o(t_f)$. We refer to this problem with a specified final fuel mass as the minimum fixed-fuel problem.

When solving for climb-to-orbit without specifying the final orbital inclination or right ascending node, the minimum fuel problem reduces to one of only three states: E, m_h, and m_o. This assumes a planar climb to orbit. This is not an unrealistic assumption when considering that a plane flying to orbit would maneuver itself into the desired orbital plane as soon after takeoff as safety requirements permit. The remainder of the flight, approximately Mach 0.5 to 25, would be mostly in the vertical plane with no significant bank angle. This allows us to ignore the position states. The three remaining states are:

$$\dot{E} = (T_x - D)\frac{V}{m}$$

$$\dot{m}_h = u_{h_a} + u_{h_r}$$

$$\dot{m}_o = u_{o_r}$$

Thus for climb to orbit, the energy-state Hamiltonian takes the simplified form. Note that the adjoints act as weighting on the type of fuel used:

$$H_a = \left[\frac{u_h + u_o + \lambda_{m_h} u_h + \lambda_{m_o} u_o}{(T_x - D)\frac{V}{m}} \right]$$

For climb to orbit, the rate of change of energy is always greater than zero. Thus, the optimal controls are found by determining the minimum of H_a with respect to the controls π_a, π_r, and h:

$$\min [H_a(\pi_a, \pi_r, h)] \; \dot{E} > 0$$

where α and δ are determined by the vertical force and pitching moment equilibrium equality constraints.

Thus, the climb-to-orbit-solution differential equations are:

$$\dot{E} = (T_x - D)\frac{V}{m}$$

$$\dot{m}_h = u_{h_a} + u_{h_r}$$

$$\dot{m}_o = u_{o_r}$$

$$\dot{\lambda}_h = H_a\left(\frac{V}{m^2}\right)\left[T_x - D + (m_h + m_o)C_p\right]$$

$$\dot{\lambda}_o = H_a\left(\frac{V}{m^2}\right)\left[T_x - D + (m_h + m_o)C_p\right]$$

where

$$C_p = \left\{T_{x\alpha} - [\,D_\alpha + D_\delta(\frac{M_\alpha}{M_\delta})\,]\right\}\left\{\frac{-F_{vm_h}}{F_{v\alpha} - F_{v\delta}\frac{M_\alpha}{M_\delta}}\right\}$$

The initial and final conditions on the adjoints are given by the transversality conditions:

$$\left[\lambda_h dm_h + \lambda_o dm_o\right]_{t_0}^{t_f} = 0$$

The values of the adjoints depend on the specific problem to be solved. The fixed-fuel-problem solution minimizes the fuel used to reach orbit without exceeding specified final fuel masses.

In this problem the vehicle has been designed and the vehicle mass and the fuel masses are fixed. The initial design may or may not have been obtained with Euler-Lagrange optimization tools. The goal is to minimize the fuel used without exceeding the specified fuel weights. If the Euler-Lagrange method reduces the fuel usage, then the onboard fuel safety margins can be increased. A similar problem maximizes the final energy for given amounts of fuel.

For the purpose of clarifying the discussion, let us assume that it is the mass of hydrogen that is fixed at the final time. Note that the approach is identical for fixed final oxygen mass; simply replace m_h with m_o and vice versa below. The minimum fuel optimization problem for fixed final-hydrogen mass takes the form:

minimize

$$[H_a(h, \pi_a, \pi_T)]_{E>0}^{\cdot}$$

Initial Conditions:

$$E(t_0) = E_0,\ m_h(t_0) = m_{h0},\ m_o(t_0) = m_{o0}$$

$$\lambda_{m_h}(t_0) = \text{free},\ \lambda_{m_o}(t_0) = \text{free}$$

Final Conditions:

$$E(t_f) = E_f,\ m_h(t_f) = m_{h_f},\ m_o(t_f) = \text{free}$$

$$\lambda_{m_h}(t_f) = \text{free},\ \lambda_{m_o}(t_f) = 0$$

From the transversality equations we get a final condition on the oxygen fuel adjoint. If a state is not fixed at the final time, then the corresponding adjoint must go to zero at t_f. Thus, we have a two-point boundary value problem. The initial states are known, and the two initial adjoint values are unknown. At the final time, two states and one adjoint are known, leaving one state and one fuel adjoint free.

The exact solution for the climb-to-orbit problem requires iterating on the initial values of the fuel adjoints to meet the final conditions. The solution's steps include the following.

1. Pick initial values of fuel adjoints $\lambda_{m_o}(t_o)$ and $\lambda_{m_h}(t_o)$.
2. Search for minimum of H_a with respect to controls (π_a, π_r, h) for current states.
3. Compute α and δ from vertical and moment equilibrium.
4. Increment motion and adjoint differential equations to next time point.
5. Stop integration at final energy E_f.
6. Compute misses on $m_h(t_f) = m_{h_f}$ and $\lambda_{m_o}(t_f) = 0$.
7. Update fuel adjoint guesses using final misses.
8. Repeat until misses approach zero.

The iterative process is straightforward and could be automated easily. However, a 3-D search on H_a, which is computationally intensive is required to find the controls; the 3-D search together with the need for iteration makes this exact solution unlikely as a candidate for onboard, real-time guidance. Therefore, we looked at approximations to the above problem that eliminate the need to iterate on the adjoints and reduce the order of the search for the optimal controls.

Approximate Solution Methods for the Climb-to-Orbit Problem

Following is a discussion of approximate solution methods to the climb-to-orbit two-point boundary value problem.

Method for approximating the fuel adjoints

The exact solution to the two-point boundary value problem requires that the optimal controls are used to forward-integrate the equations of motion and the adjoints are adjusted until the solution converges.

The values of the adjoints act as weighting on the fuel type used. To decrease hydrogen usage, the hydrogen fuel adjoint must be increased. The additional weighting reduces future hydrogen use. The iterative process required to solve for the fuel adjoints can be automated to update the adjoints using the predicted deficiency or excess of a particular fuel. For example, assume that during the low-speed flight regime, the air-breathing engine burns more hydrogen than expected. To compensate in flight for this deficiency of hydrogen, we can increase the hydrogen adjoint, as follows:

$$\lambda_{m_h} = \lambda_{m_h} + \frac{\partial \lambda_{m_h}}{\partial m_h} m_{h_f}^{\text{predicted}}$$

Similarly, for a deficiency of oxygen, the adjoint correction takes the form

$$\lambda_{m_o} = \lambda_{m_o} + \frac{\partial \lambda_{m_o}}{\partial m_o} m_{o_f}^{\text{predicted}}$$

where $m_{h_f}^{\text{predicted}}$ is the final mass of hydrogen and $m_{h_f}^{\text{predicted}}$ is the final mass of oxygen.

Further approximate methods can be used to eliminate the need to directly determine the adjoints. Since the fuel adjoints act as weighting on the type of fuel used, the net effect of

changing the value of a fuel adjoint is to change the point at which the rocket engine comes on. For example, increasing the hydrogen fuel adjoint results in the rocket engine coming on earlier than in the nominal case so that more oxygen and less hydrogen are burned. A simple method for approximating the fuel adjoints is to directly compute the energy level at which the rocket comes on. As in the previous method, the energy level for rocket ignition is determined using predicted final fuel masses:

$$E_r = E_r + k\, m_{h_f}^{\text{predicted}}$$

When using this method, it is no longer necessary to perform the search for the rocket throttle. The rocket comes on when the vehicle reaches the specific energy E_r. Thus, we have reduced the 3-D search for the optimal controls to a 2-D search on altitude and air-breathing throttle. By neglecting the small differences in λ_{m_h} and λ_{m_o}, such that $\lambda_{m_h} \approx \lambda_{m_o}$, the adjoints can be factored out of the Hamiltonian. Thus, it is no longer necessary to compute and integrate the fuel adjoint differential equations. The optimization problem now has the form

$$\min_{h,\pi_a} \left[\frac{(u_h + u_o)}{(T_x - D)\frac{V}{m}} \right]_{\substack{\dot{E}>0 \\ \pi_r = \pi_r^*}}$$

Approximate methods for finding the minimum of the Hamiltonian

The exact optimal climb-to-orbit solution requires a 3-D search of the Hamiltonian to find the optimal controls (π_a, π_r, h) for the current time. This can be computationally very demanding. The approximate methods in Figure 4 for determining the optimal controls are aimed at reducing processing requirements.

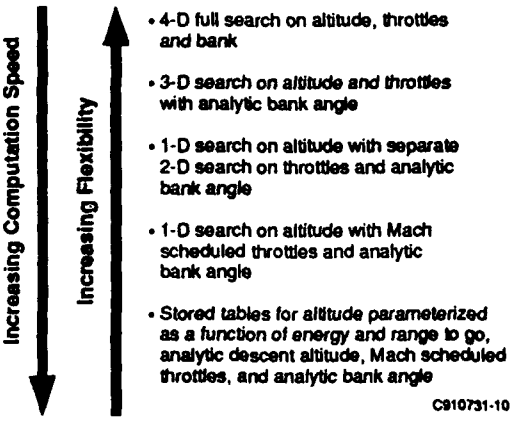


Figure 4. Methods for Finding the Minimum of the Hamiltonian

Either the second or third approach in Figure 4 is recommended as the onboard guidance scheme for determining the controls at the current time. Both these schemes closely approximate the exact optimal solution, and they can be easily implemented for real-time guidance with today's processing capabilities. The stored table approach is proposed for fast-forward solutions.

Summary of Climb-to-Orbit Guidance Approach

The climb-to-orbit guidance approach is summarized in Figure 5. The fast-forward solution integrates a model of the equations of motion ahead in time to predict the final fuel masses. The controls for the forward integration are determined from parameterized tables. The predicted final fuel masses are considered and a go/no go decision is made. The final fuel masses are used to adjust the rocket on point, and the optimal controls (h and π_a) for the current state are determined from two 1-D searches of the Hamiltonian.

Figure 6 is an example of a climb-to-orbit trajectory using this guidance approach. In this example, the vehicle starts with nominal fuel conditions at Mach 3 with enough fuel to reach orbit under nominal conditions. But between Mach 3 and Mach 12, the vehicle burns 7% more hydrogen than anticipated. Thus, at Mach 12, the vehicle has a deficiency of hydrogen. Without compensation, the vehicle would run out of hydrogen at a velocity of 20,700 ft/s. Instead, the guidance algorithm begins forward-integrating to quantify this deficiency. The prediction is used to change the rocket turn-on point. This predictor-corrector scheme is repeated until the rocket ignites. Some of the excess oxygen was used to compensate for the loss of hydrogen, and the vehicle makes orbit without running out of hydrogen. The plot in the lower right corner of Figure 5 shows the CPU seconds required per 5-sec time step. The spikes indicate when forward integrations were performed. We see that the fast-forward integration takes on the order of 2 sec of CPU† time and, therefore, is performed in real time for the GHAME vehicle model (Reference 4) used in the simulation.

Results from an error and sensitivity analysis using optimal guidance for climb-to-orbit are presented in Appendix B.

SOLUTION FOR FLIGHT-TO-LANDING-SITE

The solution for flight to specified landing site is described. Exact and approximate solutions are developed. The goal is to start at a given latitude, longitude, and energy and fly to a specified latitude, longitude and energy while satisfying some performance criterion. This problem differs from the climb-to-orbit problem in that, in addition to the fuel masses and energy, the position states, θ and ϕ, are also specified at the final time. This is the most general of the energy-state minimum-fuel problems. The solutions to this problem are used in several flight modes including sub-orbital flight to a specified landing site, return from orbit, and powered and unpowered aborts.

Exact Energy-State Solution for Flight-to-Landing-Site

The general energy-state Euler-Lagrange solution for minimum-fuel flight to landing site is summarized as follows. The performance integral J is:

$$J = \int_0^{t_f} (u_h + u_o)\, dt$$

and terminal conditions are:

E_f	final energy	final masses
θ_f	final longitude	$m_h(t_f) \geq 0$
ϕ_f	final latitude	$m_o(t_f) \geq 0$

†All CPU run times listed are the results of simulations performed on a Sun 3/60 computer with a 68020 processor and a 68881 floating point co-processor.

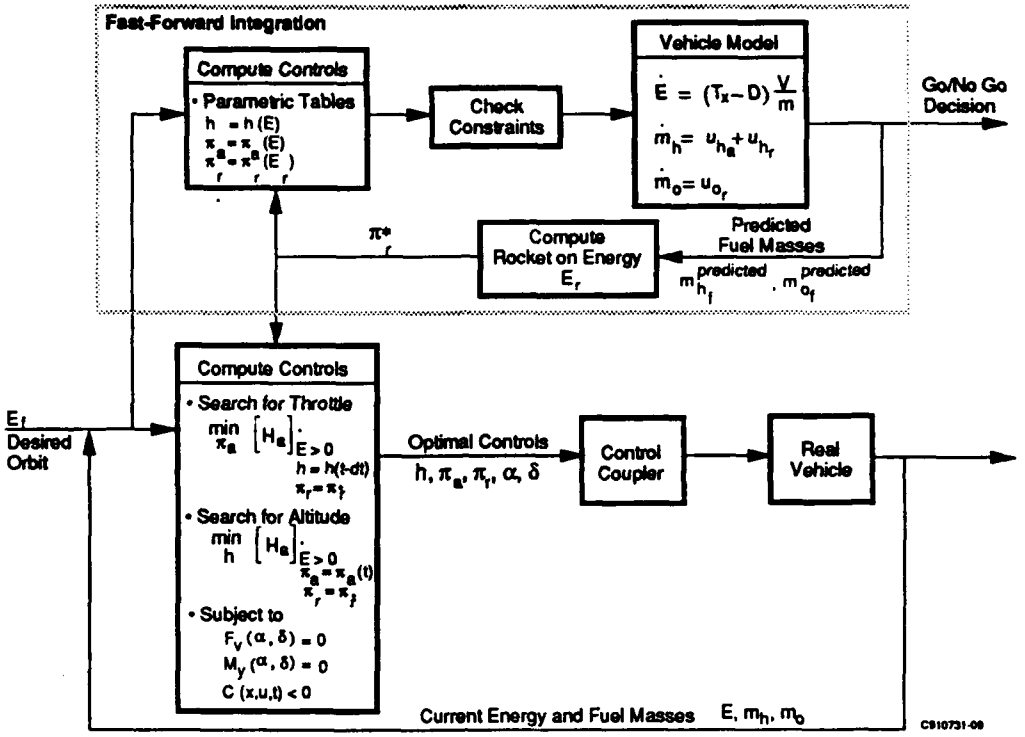


Figure 5. Summary of Climb-to-Orbit Guidance Approach

The optimal controls are found by determining the extrema of the Hamiltonian, H_a, with respect to the controls, u, and subject to the vertical force and pitching moment equality constraints:

$$u^* = \min [H_a(h, \sigma, \pi_a, \pi_r)]_{\dot{E} \geq 0}$$

$$u^* = \max [H_a(h, \sigma, \pi_a, \pi_r)]_{\dot{E} < 0}$$

subject to

$$F_v(\alpha, \delta) = 0$$

$$M_y(\alpha, \delta) = 0$$

$$C(x,u,t) \leq 0$$

The Hamiltonian has the form

$$H_a = \Big[u_h + u_o + \lambda_\theta \frac{V \cos\psi}{R_0 \cos\phi} + \lambda_\phi \frac{V \sin\psi}{R_0} + \lambda_\psi \Big(\frac{\bar{g}}{V} \tan\sigma - \frac{V}{R_0} \cos\psi \tan\phi \Big) + \lambda_{m_h} u_h + \lambda_{m_o} u_o \Big] / \Big[(T_{ax} + T_{rx} - D) \frac{V}{m} \Big]$$

One extremum operation is selected at the start; then the extremum operations are switched when

$$\max (H_a)_{\dot{E} < 0} = \min (H_a)_{\dot{E} \geq 0}$$

The adjoints are summed up as follows:

$$\lambda_\theta = C_1$$

$$\lambda_\phi = C_2 \sin\theta - C_3 \cos\theta$$

$$\lambda_\psi = C_1 \sin\phi + (C_2 \cos\theta + C_3 \sin\theta)\cos\phi$$

where

$$C_1 = \cos\phi_f \cos\psi_f C_0$$

$$C_2 = (\sin\theta_f \sin\psi_f - \cos\theta_f \sin\phi_f \cos\psi_f) C_0$$

$$C_3 = (-\cos\theta_f \sin\psi_f - \sin\theta_f \sin\phi_f \cos\psi_f) C_0$$

The constant ψ_f is the final heading angle and C_0 is related to the cruise minimum-fuel flow rate (Reference 2). The values of the two adjoint constants are unknown. The fuel adjoint differential equations are the same as for the climb-to-orbit solution. The initial values of the fuel adjoints are unknown and the final values depend on the optimization problem being solved.

In summary, the steps in the exact iterative solution for flight to landing site are as follows.

1. Choose starting operation (i.e., maximize or minimize H_a).
2. Pick initial values of adjoint constants (ψ_f, C_o) and fuel adjoints λ_{m_o} and λ_{m_h}.
3. Compute minimum of Hamiltonian with respect to controls (π_a, π_r, h, σ).
4. Compute angle of attack and elevator from vertical force and moment equilibrium.
5. Increment equations of motion and adjoint differential equations to next time point.
6. If necessary, switch optimizing operations when they are equal.
7. Stop integration at final energy E_f.
8. Compute misses on final conditions (i.e., specified position, fuel masses, adjoints).
9. Update adjoints using final misses.
10. Repeat until misses are small.

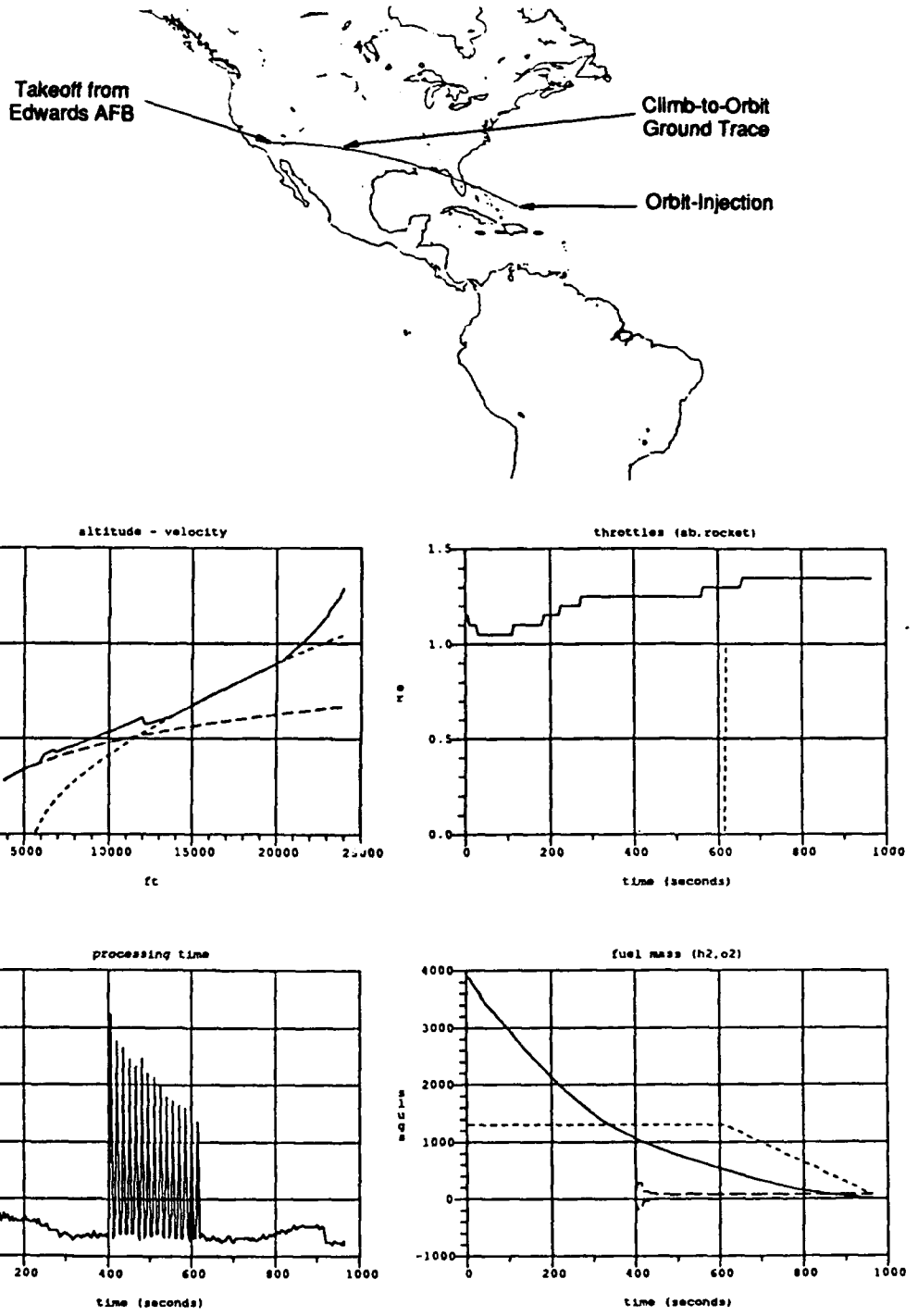


Figure 6. Results for Climb-to-Equatorial Orbit

In the iterative solution, ψ_f, C_0, λ_{m_h}, and λ_{m_o} are selected at the start. The trajectory is then integrated to the end. The solution, however, is sensitive to ψ_f. If ψ_f is not correctly chosen, the trajectory diverges near the end. A method to reduce this sensitivity is:

if

$$|\lambda_\psi(t)| \geq |\lambda_\psi(t-\tau)|$$

then

$$\sigma = 0$$

Approximate Solution Methods for Flight to Landing Site

The following sections discuss various approximations for solving the Euler-Lagrange two-point boundary value problem resulting from the exact optimal solution. Simplifications are made for the adjoint determination, the climb/descent switching logic, and the minimization of the Hamiltonian.

Approximations to the adjoints

Simple approximate solutions can be obtained for the position, heading and fuel adjoints. Adjoint constants C_0 and ψ_f can be approximated as feedback functions of range to go and heading error, respectively. The fuel adjoints can be approximated, as in the climb-to-orbit problem, as functions of the predicted final fuel masses where the final fuel masses are determined by fast forward integration.

Climb/descent switching approximation

Logic to select the starting operation (maximizing or minimizing H_g) can be approximated using relative geometry of the desired endpoint to the current vehicle position and energy. The method results in defining three flight regimes: initial descent, climb, and final descent. The three regimes and switching lines are shown in Figure 7.

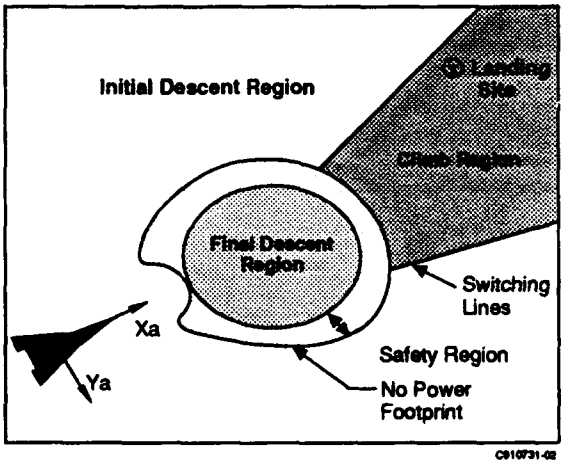


Figure 7. Climb/Descent Switching Regions

Approximate methods for finding extremum of the Hamiltonian

The exact optimal solution requires a 4-D search of the Hamiltonian to find the current optimal controls (h, σ, π_a, π_r). Various methods are available to simplify this. They were presented above in Figure 4. During two of the flight regimes, initial descent and final descent, analytic expressions for the controls can be derived. Thus, searching the Hamiltonian is required only during climbing flight.

For detailed derivations of the above approximations the reader is referred to references 1, 2, 3, 5, and 6.

The vertical dynamics that were neglected in the energy-state solutions are accounted for in the onboard system by a command coupler. The command coupler controls the vehicle to the computed optimal altitude. There are a number of possible options for a control variable. These are altitude, density, dynamic pressure, and axial acceleration (the Space Shuttle approach). Density and dynamic pressure are difficult to measure. Axial acceleration is useful during no-power descents, but is not useful for powered climbs. Altitude is chosen because it is directly measurable by the inertial navigation system, and it can be used in climb and descent.

The altitude and cross-range control laws are:

$$a_{nc} = \left(\frac{L - T_{ev}}{m}\right)\cos\sigma = \bar{g} + k_h(h - h_c) + k_{\dot{h}}(\dot{h} - \dot{h}_c)$$

$$a_{yc} = \bar{g}\tan\sigma$$

where a_{nc} and a_{yc} are nongravitational accelerations. The altitude differential equation with the controller included is:

$$\ddot{h} - \frac{D}{mV}\dot{h} = k_h(h_c - h) + k_{\dot{h}}(\dot{h}_c - \dot{h})$$

Summary of Flight-to-Landing-Site Guidance Approach

The exact optimal solution is computationally very complex and, therefore, not recommended for onboard implementation. Several approximate methods are better suited for onboard guidance. The stored parametric table approach is recommended for the fast-forward solutions, because it has low computation requirements. The search solution for the climb controls is recommended for guidance because it is flexible to changes in models and low fuel usage.

The flight-to-landing-site processing elements are: (1) current optimal control processing, and (2) fast-forward solution to compute final fuel weights. These are shown in Figure 8.

The current "optimal control processing" box in Fig. 8 is shown in detail Figure 9. The controls for the fast-forward solutions are computed in the same way as the current optimal controls, except during climb where pre-stored parametric tables are used.

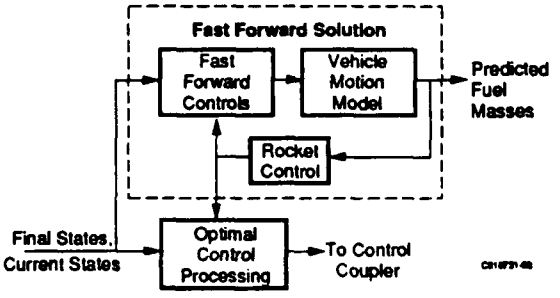


Figure 8. Flight-to-landing-site Processing

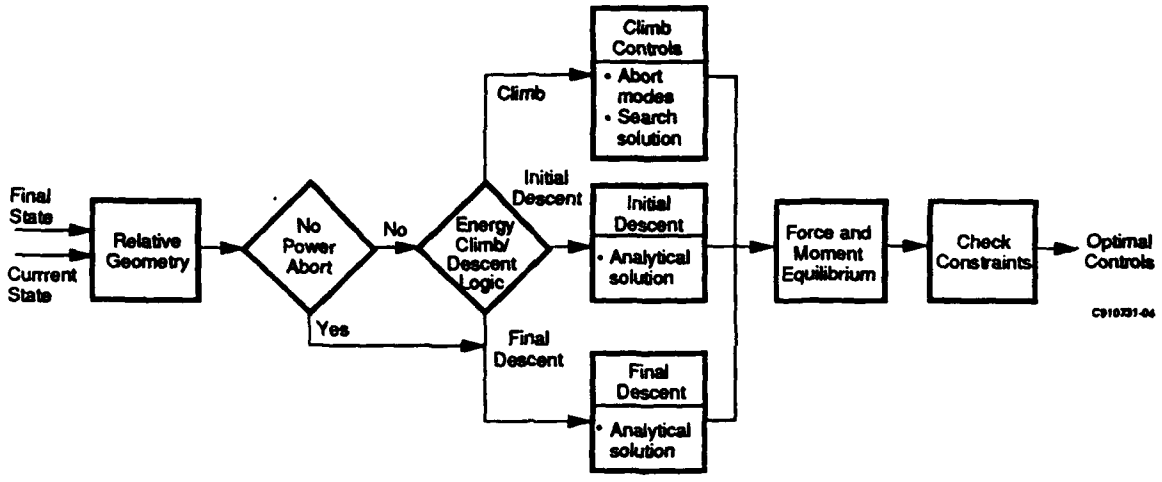


Figure 9. Processing for Computation of Current Optimal Controls for Flight-to-Landing-Site Mode

The inputs to the guidance processing are the desired final state of energy, latitude and longitude, and the current states of mass of hydrogen, mass of oxygen, altitude, velocity, and heading. The relative geometry module computes the range and heading angle to the end point.

In the flight-to-landing-site approximate solution, the trajectory is broken into climb/descent regions, as was shown in Figure 7. In the initial and final descent regions, analytical methods are used to compute the optimal controls. In the climb region simplified search methods are used to compute the optimal controls.

This climb/descent processing logic separates the flight area into three regions: initial descent, climb region, and final descent. The final descent circular region is computed from fast-forward solution for the maximum range glide and the minimum range glide. The switching lines are computed from an analytical expression. The method for computing the controls depends on the area where the final point is located relative to the aircraft.

If the final point is inside the initial descent region, an unpowered minimum drag descent is commanded. The rocket throttle is set to zero, and the air-breather throttle is set at the minimum flame-holding condition. The altitude profile is defined by the minimum drag path with approximate vertical equilibrium, and the roll angle is computed to drive the heading error to zero.

If the aircraft is inside the climb region, a powered climb is commanded. The current optimal controls are determined from an altitude, air-breathing throttle, and rocket throttle search. If the ratio of fuel reaches that necessary to fuel the rocket, the air breather is turned off. The roll angle is computed to drive the heading angle error to zero.

At the time when the final point is selected, it is not necessarily known if there is sufficient fuel to reach that point, especially in an abort situation. Thus, a computation is made to predict if the desired end point can be reached. If it can't be reached, a new end point is selected. Different prediction methods are used depending on the type of abort. These are given in Table 3.

For no-power and rocket-power configuration, a footprint (locus of point that can be reached) can be computed. Because of the simple engine models, it is feasible to do this. For the case where the air breather only or the air breather and the rocket are operating, it is computationally difficult to compute footprints. Thus, for these cases, only a single trajectory to the selected landing site is computed. The fast-forward computation approach is shown in Figure 10.

Table 3. Prediction Methods for Different Failures

Engines Available	Prediction Method
Air-breathing Engine only	Fast forward solution to the end
Rocket engine only	Rocket footprint
Both engines operating	Fast forward solution to the end
No engines operating	No power footprint

This trajectory is computed with the same algorithms as the current optimal solution, except for the climb control solution, which is simplified to reduce computation time. Stored parametric tables for the air-breathing throttle and altitude are used. A model of the vehicle equations of motion is used in the solution.

Figure 11 shows simulation results for a three-phase descent/climb/descent trajectory. An abort was triggered at Mach 15 during a climb to a polar orbit. Following the abort, the vehicle turns around and returns to Edwards AFB.

FOOTPRINT GENERATION

A footprint defines an area of attainable final points. The Euler-Lagrange formulation of the footprint problem is one of maximizing cross range for a series of specified downranges as follows:

$$J = \int_0^{t_f} \phi \, dt \quad \text{with terminal condition on } E_f \text{ and } \theta_f$$

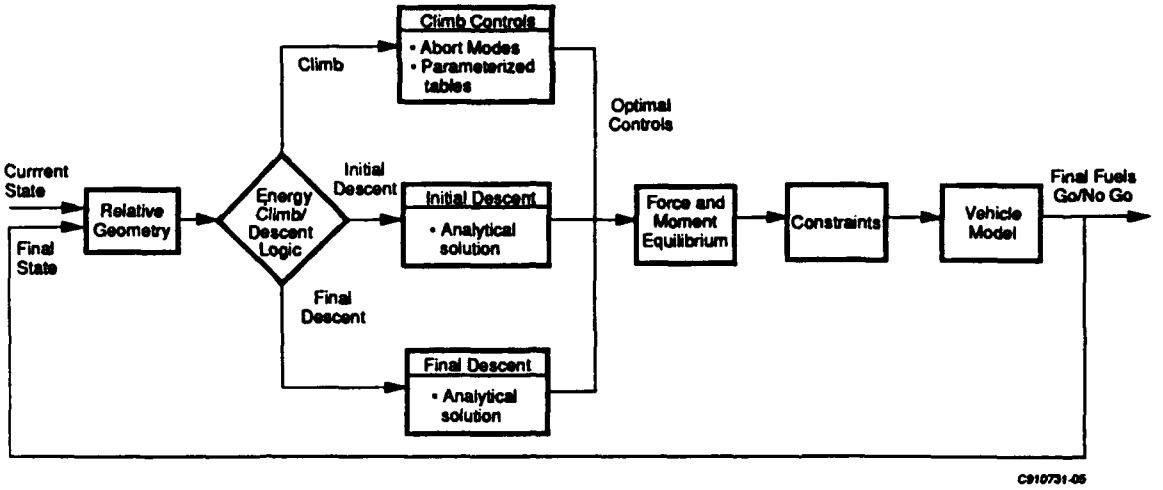


Figure 10. Fast-Forward Prediction Processing For Flight-to-Landing-Site Problem

The exact optimal solution requires iteration on one adjoint constant. For onboard real-time implementation, closed-form solutions were developed that closely approximate the optimal controls. The details of the solution are given in reference 1. The unpowered footprint is used in flight-to-landing-site guidance to define the final descent region. Rocket-powered footprints are used to determine if a chosen landing site is attainable using rocket power only with the current fuel masses.

The onboard footprint calculator is accurate for spherical earth and includes earth's rotation effect and complex constraints. The method for calculating a footprint is shown in Figure 12.

The footprint is calculated starting with the vehicle's initial altitude, velocity, heading, fuel mass, and position. The energy-state equations of motion are integrated along the commanded roll angle and drag profile until a minimum energy is reached. A variable time step depending on the magnitude of the bank-angle turn rate is used. At the minimum energy point, the final latitude and longitude define the footprint boundary. A 540-deg sweep of heading angles is conducted using minimum and maximum drag flight conditions. The roll angle control is proportional to the heading error. The trajectory termination points, 50 in all, create the closed boundary. The CPU time to create an unpowered footprint is approximately 5 to 10 sec depending on initial Mach number.

For powered footprints, it is desirable to fly the vehicle at minimum thrust when turning at high rates in order to lower the kinetic energy and thereby reduce the radius of curvature. The switching logic is similar to that developed for the flight-to-landing-site solution. This concept is illustrated in the diagrams in Figure 13.

Figure 13a shows the three phases of the rocket-only trajectory as energy losses and gains and the distance traveled in each phase. In the initial descent, the vehicle is turning at a high rate in order to align itself with the commanded heading. Once the vehicle heading nears this desired heading, the engine is switched on and the vehicle climbs until all of the rocket fuel is used, at which point a glide descent begins. Figure 13b shows the geometry involved.

The footprint for unpowered and rocket-powered flight is shown superimposed over a world map in Figure 14.

IN-FLIGHT VEHICLE GUIDANCE CONFIGURATION

An in-flight guidance, navigation and control structure derived from the optimization solution is shown in Figure 15. The system configuration combines the element of footprint generation, fast-forward prediction, and current control processing into a complete vehicle guidance configuration.

The inputs to the guidance system are: the type of mode and desired final conditions from the pilot; inertial navigation sensor position, velocity, and heading from the inertial sensors; and fuel flow rates and fuel weight from the fuel sensors. The processing functions are optimal guidance commands, fast-forward prediction, footprint calculation, and an altitude command coupler.

A fast-forward prediction module computes a trajectory with the air-breathing engine operating to the final point to determine if there is sufficient fuel to reach the point. The footprint generator finds the area of attainable final points for unpowered and rocket-only flight.

The optimal guidance command module computes the current optimal air-breathing throttle, rocket throttle, altitude, and roll angle commands. These are output to the command coupler and the engine controllers.

The optimal acceleration and roll angles commands are sent to an attitude controller.

CONCLUSIONS

The Euler-Lagrange energy-state method is shown to be a powerful tool in solving the in-flight vehicle guidance optimization problem. For in-flight guidance, the Euler-Lagrange energy-state optimization techniques have been extended to the hypersonic regime and 3-D turning flight.

The approximate approaches give trajectories very close to the full optimal solutions for minimum fuel. However, the computing time is much lower than for the full optimal iterative solutions. Thus, the approximate Euler-Lagrange optimization method can be used on-board for mission guidance. This approach provides for high flexibility to mission changes, less operator involvement, and less launch-site support for pre-mission trajectory generation. Approximate solution methods were developed for the climb-to-orbit problem and the flight-to-landing-site problem.

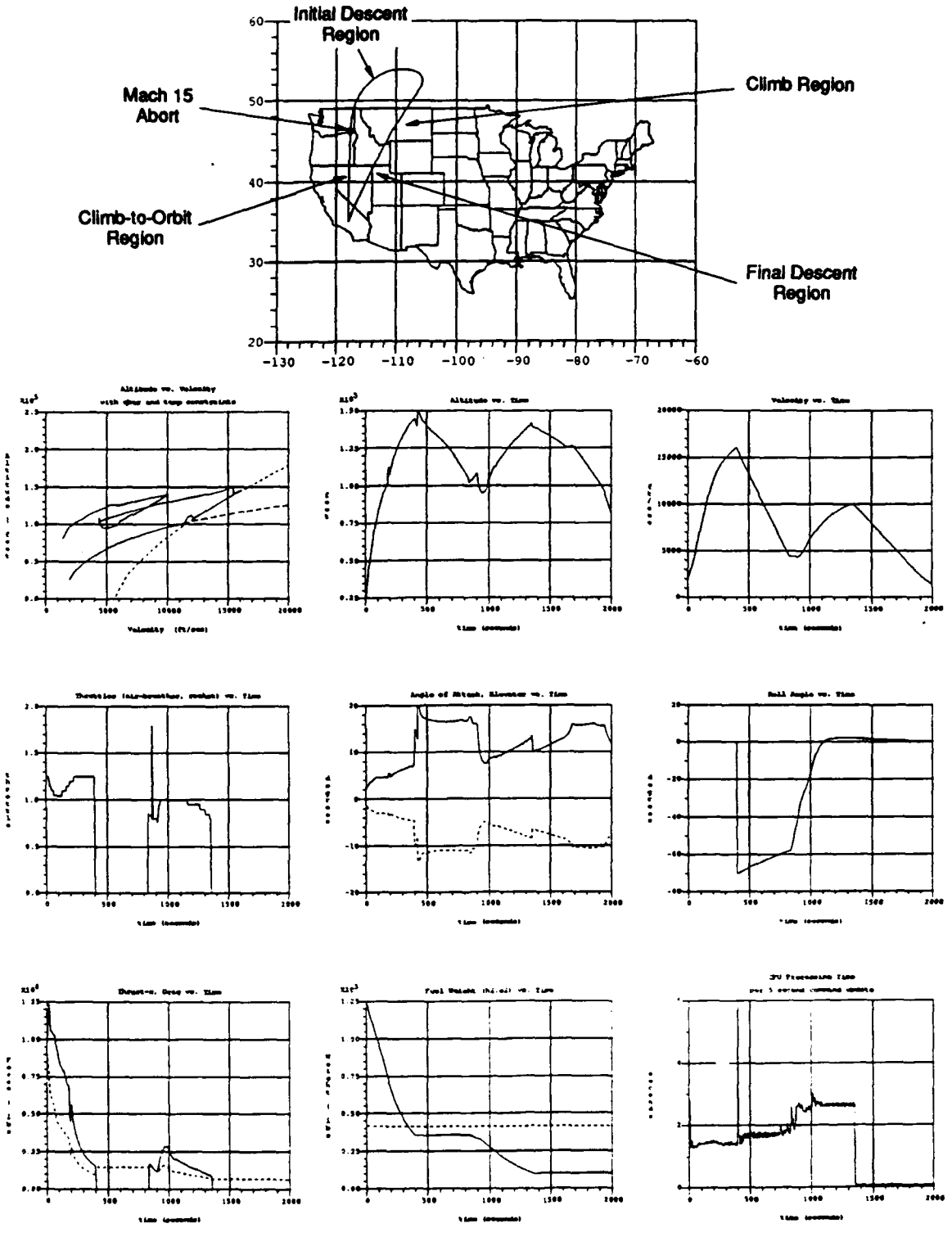
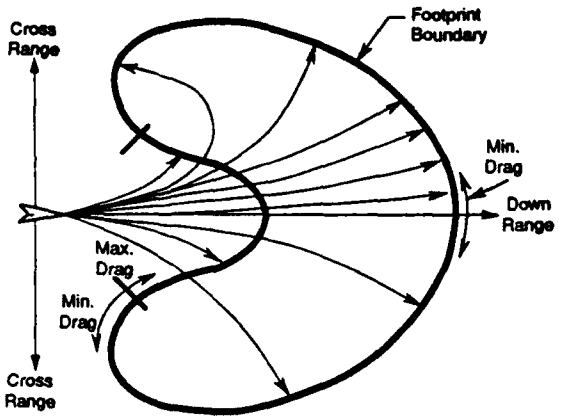


Figure 11. Simulation Result: Descent/Climb/Descent Trajectory Mach 15 Powered Return to Edwards AFB



Footprint Calculation Method

- Input velocity, altitude, mass of fuel, heading, and position
- Iterate heading command, Ψ_{com}

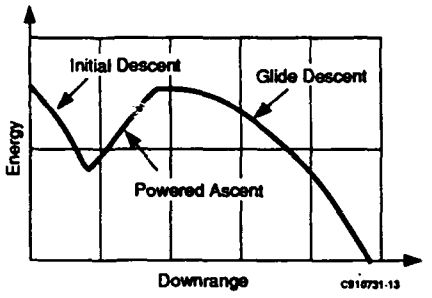
$$\tan(\phi) = (\Psi_{com} - \bar{\Psi})^2$$

$$\bar{\Psi} = \frac{\tan(\phi)}{V}\left[g - \frac{V^2}{R}\right]$$

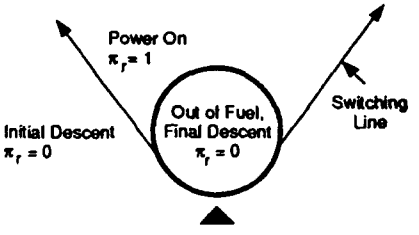
- Set drag to min or max
- Variable time step for integration of equations of motion
- Stop integration when minimum energy is reached
- Footprint boundary defined by 50 points

C910731-08

Figure 12. Footprint Generation Method



a. Three-Phase Footprint Trajectory Profile



b. Switching Line Geometry

Figure 13. Three-Phase Powered Footprint Trajectory Diagrams

In the climb-to-orbit solution, fast-forward integration is used to predict final hydrogen and oxygen masses. The fuel masses are used to provide information for a go/no-go decision and for control optimization. The optimal guidance adjusts the trajectory to account for fuel mass changes due to vehicle uncertainties. The optimal controls are determined from a simplified search method.

An error and sensitivity analysis for the climb-to-orbit problem shows that the optimal control results in more fuel margin because the trajectory is adjusted using the predicted hydrogen or oxygen mass (e.g., if the hydrogen fuel is projected to be low, the rocket is turned on sooner so that less hydrogen and more oxygen is burned, so that all of the available fuel is burned, whereas for the stored profile case the mission is aborted if either fuel type is predicted to be exhausted). Thus, it is important in the early vehicle design studies to include trajectory optimization and guidance analysis because the ability of the guidance system to accommodate for uncertainties greatly affects the magnitude of the fuel margins required.

In the flight-to-landing-site approximate solution, the flight is broken into climb and descent regions. In the descent regions, analytical methods are used to compute the optimal control. In the climb region, simplified search methods are used to compute the optimal controls. Fast-forward integration to a landing site or footprint calculations are used to answer the important question of whether there is sufficient fuel to fly to the selected landing site. The fast-forward solution method determines the climb controls from a pre-stored parametric table of the controls as functions of range and Mach number. In the event of mission abort with the air-breathing engine still operating, the flight-to-landing-site solution is modified depending on the type of configuration available.

A computationally efficient method for generating rocket-powered and unpowered descent footprints is also described. Footprints are used in defining the three flight regimes under normal flight conditions. Under air-breathing engine failure or other emergency conditions, footprints are also useful for quickly identifying candidate landing sites.

REFERENCES

1. Schultz, R.L., Hoffman, M.J., Case, A.M., Sheikh, S.I. *Hypersonic Vehicle Trajectory Algorithms (HVTA); Final Report*, Wright Laboratory, WPAFB, Ohio, 45433-6553, Contract Number F33615-86-C-3604, 30 April, 1991, to be published.

2. Vinh, N.X., *Optimal Trajectories in Atmospheric Flight*, (Amsterdam: Elsvier, 1981).

3. Schultz, R.L., and Zagalsky, N.R., "Aircraft Performance Optimization for Aircraft," *Journal of Aircraft*, Vol. 9, No. 2, February 1972, pp. 108–114.

4. Bowers, A.H., and Iliff, K.W., *A Generic Hypersonic Aerodynamic Model Example (GHAME) for Computer Simulation*, NASA Ames–Dryden Flight Research Facility, August 3, 1987.

5. Schultz, R.L., "Three-Dimensional Trajectory Optimization for Aircraft," *Journal of Guidance, Control and Dynamics*, to be published.

6. Schultz, R.L., Hoffman, M.J., Case, A.M., Sheikh, S.I. "Navigation, Guidance and Trajectory Optimization for Hypersonic Vehicles," *AGARD Hypersonic Combined Cycle Propulsion Conference Precedings No. 479*, Madrid, Spain, June 1990, Paper No. 3.

7. Graves, C.A. Jr., and Harpold, J.C., *Shuttle Entry Guidance American Astronautical Society 25th Anniversary Conference Proceedings*, Paper AAS78-147, Houston, October 30 to November 2, 1978.

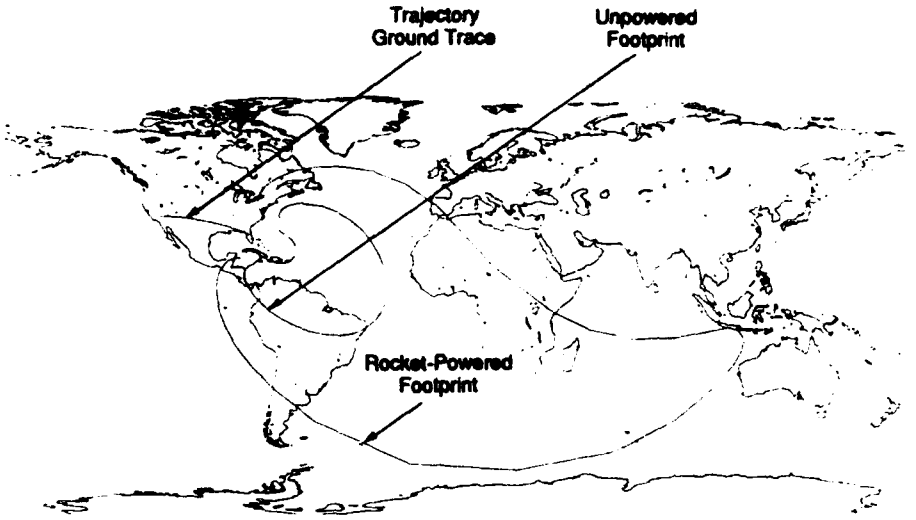


Figure 14. Footprints For Unpowered and Rocket-Powered Flight

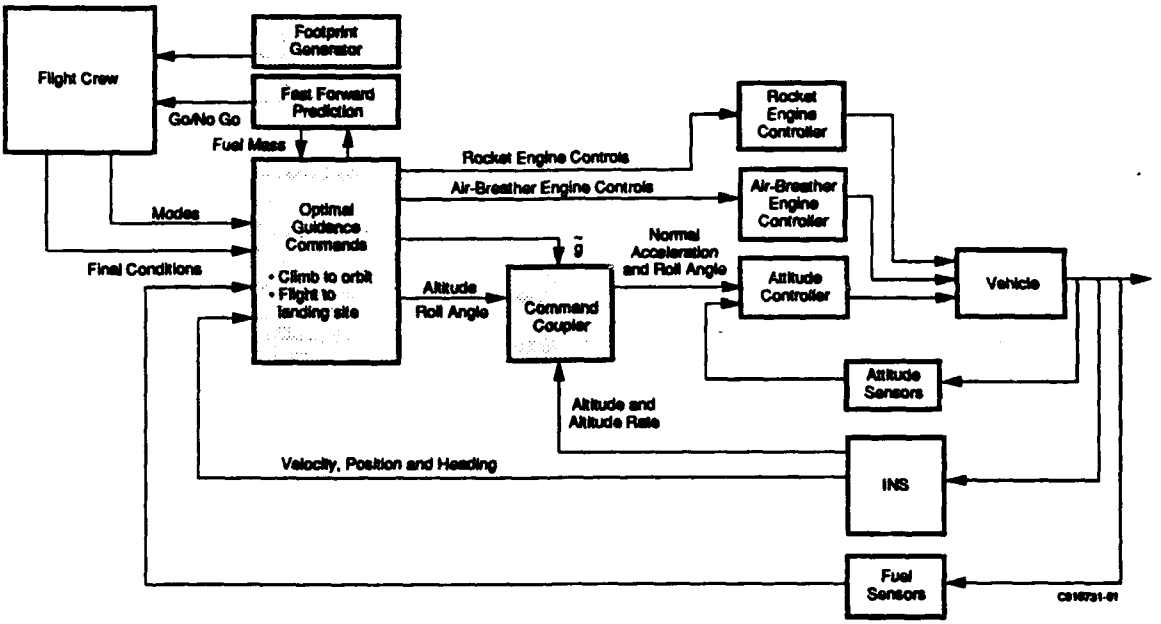


Figure 15. Guidance Navigation and Control Configuration

APPENDIX A—CANDIDATE OPTIMIZATION METHODS

The candidate optimization solution approaches and a brief comparison of their capabilities are shown in Table A-1.

The nonlinear programming methods are suitable for off-line computation but at the present time are too complex for onboard implementation. The best candidate for real-time hypersonic vehicle trajectory optimization is the Euler-Lagrange reduced-order-models method.

APPENDIX B - ERROR AND SENSITIVITY ANALYSIS FOR CLIMB TO ORBIT

The objective is to determine the likelihood of the vehicle attaining orbit given a set of initial conditions and uncertainties. A Monte Carlo study was performed. In the Monte Carlo analysis, perturbation values are randomly selected for each performance parameter based on chosen 1σ values, and a climb-to-orbit trajectory is flown using these values. The final velocity achieved for this configuration of parameter variations is recorded, and then a new set of random perturbations is chosen for the next trajectory. This is repeated until sufficiently many trajectories have been generated in order to provide a good idea of the probability of achieving orbital velocity. In this study 500 trajectories were generated in each run.

Table B-1 shows the 1σ values used in this study. These selections of sigma values are based on considerations of the models, previous studies, GRAM (Global Reference Atmosphere Model), and initial results. The selections are partly arbitrary and are certainly not meant to be the final word concerning the 1σ values.

Table A-1. Candidate Optimization Solution Approaches

Approaches	Comparison
• Euler-Lagrange	• Provides general solution but results in two-point boundary value problem (TPBVP)
– Newton-Raphson – Reduced order models – Singular perturbations	– Has sensitivity problems – Simplifies the TPBVP – Gives general method for model reduction
• Gradient methods – Steepest descent methods	• Is computationally complex and can find local minimums
• Nonlinear programming – Conjugate gradient – Feasible directions – OTIS	• Can solve full state optimization problems and handle complex constraints but computationally complex

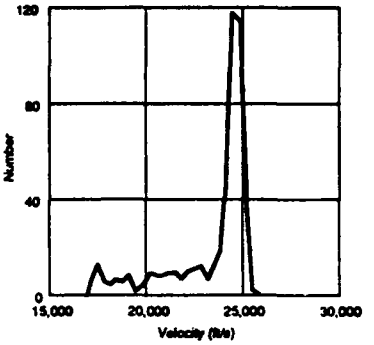
Table B-1. 1σ Values

Parameter	1σ Error
Drag	±5%
Specific Impulse	±6%
Empty Vehicle Weight	±5%
Air Density	±7%
Speed of Sound	±4%
Winds	±50 ft/s

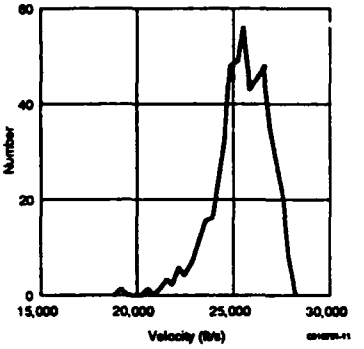
Three Monte Carlo runs for fuel margins of 0, 10, and 20% for both the pre-stored and the optimal trajectories were completed. The final velocity distributions for each run were recorded and used to construct the probability density functions.

Figure B-1a shows the probability density plot for the pre-stored algorithm with 10% fuel margin. It compares the number of occurrences of the climb-to-orbit trajectory reaching each final velocity. Upon inspection we see that the probability of the trajectory reaching orbit (~25,000 ft/s) is slim. This plot can be compared with the probability density results of the optimal algorithm with 10% fuel margin shown in Figure B-1b. In this plot a much more favorable distribution is seen.

Using the data in Figure B-1 one can calculate the probability distributions shown in Figures B-2. These show the percentage likelihood of reaching any final velocity for the different algorithms and initial fuel margins. For instance, in the pre-stored case we see that with zero safety margin, the climb-to-orbit has approximately a 4% chance of reaching orbital velocity, while the optimal guidance trajectory with zero safety margin has a 50% chance of reaching orbit.

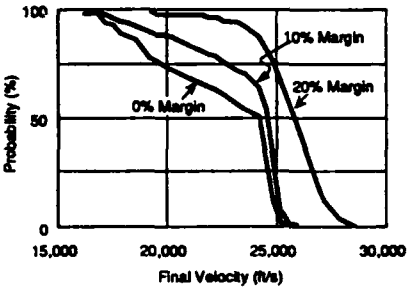


a. Pre-stored Trajectories with 10% Fuel Margin

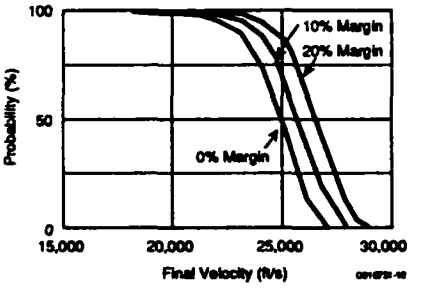


b. Optimal Trajectories with 10% Fuel Margin

Figure B-1. Probability Density of Final Velocities



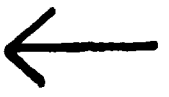
a. Pre-stored Trajectories



b. Optimal Trajectories

Figure B-2. Probability Distribution of Final Velocities

The sensitivity analysis was completed using the GHAME vehicle (reference 4). The value of using optimal trajectories has been demonstrated. In this analysis. if the nominal trajectory contains a 10% safety margin, the in-flight optimized trajectory has a 70% chance of reaching orbit, while the pre-stored trajectory has only a 20% chance.



Technology and Standards for a Common Air Mission Planner

Ellen K. Kriegel
Joseph K. DeRosa
The MITRE Corporation
Burlington Road
Bedford, Massachusetts 01730
U.S.A.

1. SUMMARY

A common air mission planning system can be designed to help create mission plans for various aircraft and weapons systems. Even though users are planning missions for different aircraft, they perform common functions, access identical databases, and operate with similar procedures. An architecture based on modern computer-communications technology and widespread standards can exploit this commonality to produce a mission planning system of enduring value. (Ref. 1).

The USAF Electronics System Division (ESD) is currently developing such a common air mission planner in a phased program for multiple aircraft and weapons. It is called Air Force Mission Support System (AFMSS) and will be used by the Strategic Air Command, Tactical Air Forces, Military Airlift Command, and Special Operations Forces of the United States.

This paper discusses the common elements that constitute the core of that mission planning system, as well as the accommodation of unique elements for each different aircraft. The open system architecture concept is established, and the technology trends and standards that drive that architecture are defined. Performance issues are discussed, and the Common Mapping System is presented. Finally, future challenges in mission planning systems are discussed.

2. ELEMENTS OF MISSION PLANNING

In generating mission plans for different aircraft, many of the functions performed, the data accessed, and the procedures used are quite similar. This section discusses the common elements (listed in Table 1) of mission planning systems for these dissimilar aircraft.

Table 1. Common Elements of Mission Planning

Common Functions	Common Databases	Common Procedures
Flight Planning	Operational Data	Manipulate Data, Maps,
Route Selection	Air Tasking Order	Imagery
Ingress/Egress	Personnel	Predict Radar/IR/Visual
Take-off and Landing	NAVAIDS	Images
Threat Penetration Analysis	Weather	Rehearse Mission (Fly Thru)
Threat Locations	Weapons Data	Initialize Aircraft Avionics
Terrain Masking	Communications Plans	Mission Material Preparation
Aircraft Visibility	MCG&I Data	
End-Game Tactics	Intel Data	
Target Visibility	Threat Locations	
Weapon Delivery	Imagery	
Deconfliction	Electronic Order of Battle	

2.1 Common Functions

There are three major categories of functions that are performed in generating a mission plan:

1. Flight planning
2. Threat penetration analysis
3. End-game tactics

These functions are common to all, or at least certain classes of, aircraft and weapons. By determining the common elements, the software architecture can be constructed to allow re-use of these elements, thus saving development effort and time. Flight planning encompasses route selection for ingress and egress as well as calculation of parameters associated with take-off and landing, maneuvers, and fuel consumption. The goal of threat penetration analysis is to determine a path of minimum danger taking into account threat locations, terrain masking, and aircraft visibility. End-game tactics are determined from calculations based on target location and visibility and on weapons parameters.

Although the performance parameters are generally different for each different aircraft and weapon, the methods of calculation are quite similar. For example, fuel calculations for different aircraft are made in identical ways using corresponding polynomials or look up tables that describe the performance characteristics of the different aircraft. Thus, it is possible to construct common software to execute the common functions with access to separate modules containing the aircraft and weapons-specific parameters.

2.2 Common Databases

There are also three common classes of data that are needed for mission plans for different aircraft:

1. Operational data
2. Mapping, cartography, geodesy & imagery
3. Intelligence data

Thus it makes sense to build these databases with common data models and to access them with common protocols.

Operational (OPS) databases contain mission information obtained from both higher echelon commands and airbase information systems. For example, in the Gulf War, the Theater Commander issued a single Air Tasking Order (ATO) for all aircraft in the theater of operation. Likewise, weather observations and forecasts are common to all aircraft over the target area; and under the new composite wing concept for the U.S. Air Force, it would be reasonable to construct a single common database at the wing to supply personnel, maintenance, and munitions information to mission planners for dissimilar aircraft. Mapping, cartography, geodesy and imagery (MCG&I) databases contain maps of varying scale, photographs, terrain elevation and features information. A list of the types of typical MCG&I products for mission planning is provided in Table 2. Intelligence (INTEL) databases contain threat locations and target imagery data, which have to be registered to the maps. The inherent size of these databases is quite large, and they utilize a large portion of storage capacity in a mission planning system.

Table 2. MCG&I Data Types

Mapping, Cartography, Geodesy, and Imagery Products	
ADRG	Arc Digital Raster Graphics (Scale Varies)
DTED	Digital Terrain Elevation Data (3 Arc-Second)
DFAD	Digital Features Analysis Data (3 Arc-Second)
WVS	World Vector Shoreline (1:250,000 scale)
DCW	Digital Chart of the World (1:1,000,000 scale)
PVOD	Probabilistic Vertical Obstruction Data
Text	Text Data (Digital Airfield Information File, Chart Update Message/Electronic Chart Update Message)
ADRI	Arc Digital Raster Imagery (typically 10m SPOT)

By standardizing the formats and methods of accessing these databases, theater-wide interoperability is ensured between Mission planning systems and their key suppliers of Command and Control information. In addition, the OPS, INTEL, and MCG&I portions of the mission planning system can now be made common.

2.3 Common Procedures

All mission planners have the need to manipulate and display data, by methods that can be standardized. Some of these procedures include manipulating maps and imagery (panning, zooming, and rapid access of different types of MCG&I data for the same area of interest), the input of external data via autofeeds and physical devices, the compartmentalization and securing of data, the presentation of visual images such as radar predictions and terrain perspective views, mission rehearsal (fly-through), and the creation of printed materials for combat mission folders.

3. OPEN SYSTEM ARCHITECTURE

3.1 Development Objectives

The development objectives for a common air mission planner are to take advantage of common functions, databases and procedures and to provide a single planner for a variety of aircraft and weapons. This single planner, with its aircraft/weapon specific modules would have interoperability through standardized network autofeeds to supply the necessary intelligence and operational data in real time. In addition, these intelligence and operational data feeds need to interface with only one mission planner.

The goal is to use commercially available products whenever possible. For example, high-end workstations are deployable to operational airbases, are competitive in price, and achieve the needed performance. By utilizing commercial standard operating systems, databases, and windows packages, and by "re-using" the common software and developing only aircraft/weapon specific modules, overall cost for providing planning for multiple types of aircraft and weapons is reduced. To continue to benefit from these advantages, however, the software architecture must readily allow the addition of new aircraft, weapons and planning functions in a way which decouples the software development cycles of the common mission planner on one hand and the aircraft specific software and avionics software on the other. In addition, the planner must support an open system architecture which allows the software to be easily rehosted to new hardware platforms as commercially available products change.

3.2 Performance Objectives

The performance objectives of the common air mission planner are to permit short planning cycles (as low as 15 minutes for some fighter aircraft) and to allow threat updates of large threat databases in near real time. Other performance drivers include the ability to perform radar predictions and rapid imagery manipulations, such as terrain perspective views in seconds and fly-through at rates of one to five frames/second. Some acceptable response times are provided in Table 3 (Ref. 2). A recent study (Ref. 3) indicated that response times of under 0.5 seconds, if they can be achieved, markedly enhanced operator productivity. The compression and display of MCG&I data is also an important performance factor in evaluating the planner.

Table 3. Mission Planning Display Response Times

Function	Response Time
2D Displays	
Map and Plan View Displays:	
Pan	5.0 seconds
Zoom	1.0 second
Profile Views	5.0 seconds
3D Displays	
Imagery	30 seconds
Radar Predictions	10 seconds
Electro-optical Predictions	10 seconds
DFAD and VOD Overlaid on DTED	1.5 seconds
Window Interactions (open, close, resize, reposition, select)	1.0 second
Route Fly-through (movie loop)	1 to 5 frames per second

This performance needs to be provided on a worldwide deployable or even portable system. It is to be used by the aircrews in the field and therefore should be user interactive, and user friendly with rapid response rates.

3.3 Elements of the Architecture

The method for achieving these development and performance objectives is to take advantage of the latest computer technology, including high-end performance graphics workstations and peripherals packaged in transit cases. A distributed client/server hybrid architecture allows flexibility and cost-effective use of peripherals and storage. A combination of POSIX compliant secure operating systems and commercially available secure databases running on a workstation provide the basis for a multilevel secure design, that protects data from unauthorized access, which can be certified as required by specific users.

Critical for making the system adaptable to new weapon and aircraft systems is the software architecture. This needs to maximize the use of common re-usable software modules and have a common interface to aircraft-unique modules of objects, that provides isolation, as shown in figure 1a. Without such an architecture the "common" air planner would, in fact, quickly become a collection of aircraft specific systems with diverging baselines running on the same hardware suite. This common software architecture is under development as part of the AFMSS contract, and MITRE is in the process of developing criteria for evaluating the proposed software architectures. Figure 1b shows the hardware configuration.

To enhance portability and interoperability, a set of standards for AFMSS were developed. Specifically a Unit Level Open System Architecture (ULOSA) was specified for system interoperability and data communications and for the system environment (e.g., operating system, windows, user interface). For MCG&I data a Common Mapping System (CMS) standard was developed. ULOSA and CMS are discussed in sections 5 and 7, respectively.

4. TECHNOLOGY TRENDS

It is the advances in computer communications technology (Ref. 4) that make meeting the technical challenges outlined in section 2 possible and allow us to build highly capable aircraft mission planning systems. Even though the standards-based architecture results in substantial performance penalties, they are becoming acceptable because of lower cost/performance ratios of hardware and improved performance of the COTS standard packages. We believe the development and performance objectives discussed in section 2 can be met with a standards-based architecture. Both the speed and storage capacity of integrated circuits are growing by about two orders of magnitude per decade

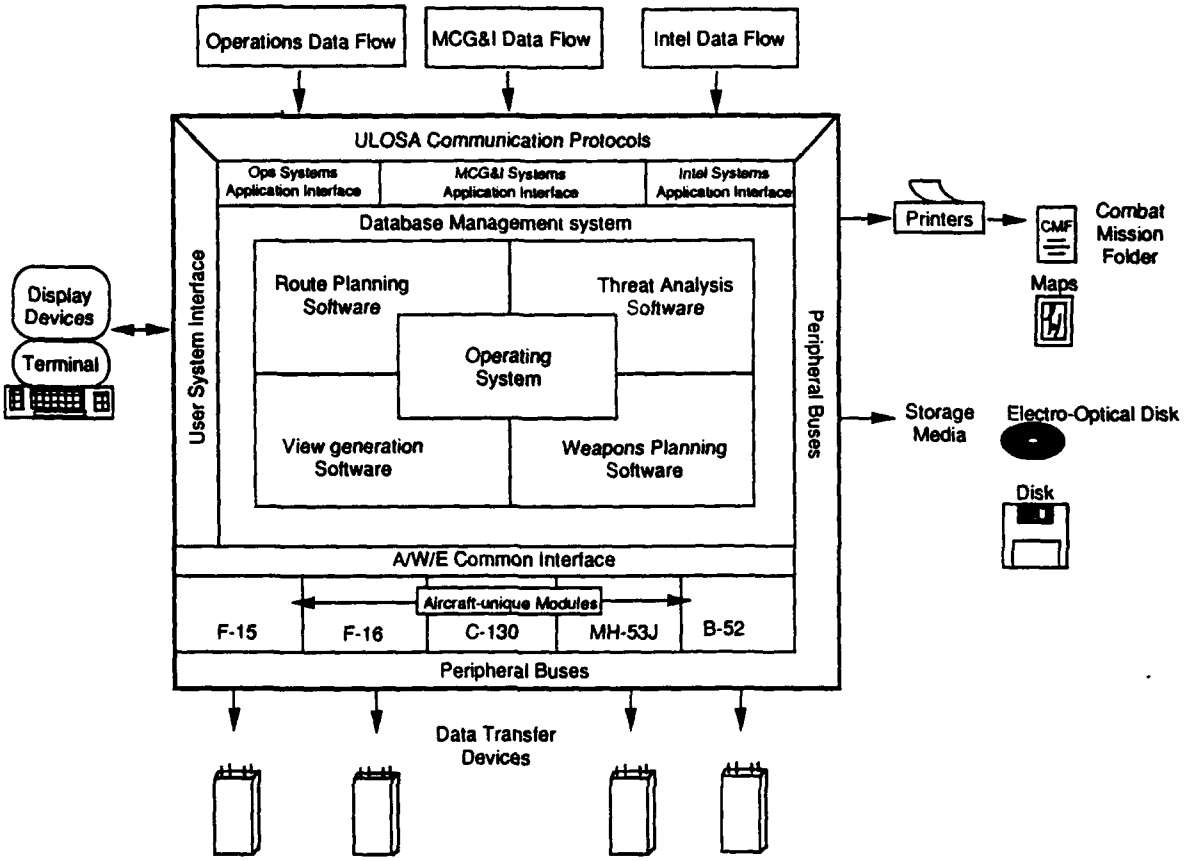


Figure 1a. AFMSS System Architecture

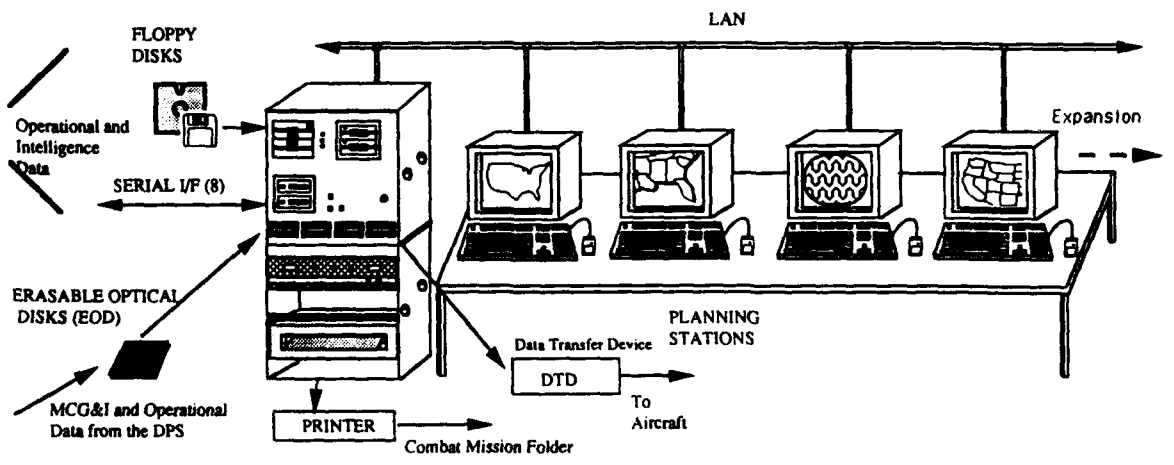


Figure 1b. AFMSS Hardware Architecture

(Ref. 5). Figure 2 (from Ref. 2) indicates that the trend in compute power will bring us to 500-1000 times that of a VAX 11/780 by the year 2000. Several popular workstations costing under U.S. $100,000 are indicated on the chart. It is the availability of these powerful, low cost off-the-shelf workstations that make rapid response times and manipulation of large amounts of data using standard packages feasible.

In addition to impressive levels of CPU power, high-end workstations provide capable graphics subsystems. These subsystems, employing coprocessors and application specific integrated circuits, provide the high drawing rates needed to speed display response times, while off-loading the main CPU from compute intensive rendering tasks. Although the hardware in these subsystems is proprietary and unique to each vendor, the graphics standards such as PHIGS+, are layered on top of the operating system device driver which connects to this proprietary hardware. Thus, the details of the hardware are hidden from the programmer, and the application program can remain portable across multivendor platforms.

Another technology driver is storage. Commercial workstations with virtual memory operating systems can support many gigabytes of on-line disk storage, employing multiple disk drives. The amount of storage provided in a system is dictated by the cost, weight, and physical size limitations of the individual drives. For example, today's state-of-the-art in 3.5-inch form factor disk drive is capable of storing over a gigabyte of data and costs approximately U.S. $5,000. Capacities per drive are doubling every 12 to 18 months with price per device remaining constant. In multi-workstation mission planning systems, on-line disk storage capacities can be leveraged by providing access to common data via a local network. At the present time, four to six gigabytes of on-line storage represent a reasonable weight, cost, performance trade-off.

The increased processing power of workstations is being matched by performance improvements in portables and laptops. Portable mission planners with MCG&I displays, image manipulations, and adherence to open architecture standards are becoming a reality, though our studies indicate that quality of maps and response times are still "marginal" (Table 4).

Standardized Local Area Network (LAN) products that support data bandwidths of a few megabits per second are readily available off-the-shelf. (Ethernet/IEEE standard 802.3 are discussed in section 5.) This permits standardizing interconnectivity between intel, ops and mission planning systems. The Fiber Data Distribution Interface (FDDI) will provide an order of magnitude improvement over Ethernet, with another order of magnitude improvement likely in the late 1990's should increased capacities be required.

It is clear that the trend in workstations is toward being faster and more capable. Strict adherence to standards will allow mission planning workstations to be upgraded, either to enhance performance, or to replace older non-supportable units without costly software rewrites. It also allows successive generations of mission planners to coexist as the migration takes place. The next section discusses the standards being adopted for the Air Force's common mission planner, AFMSS.

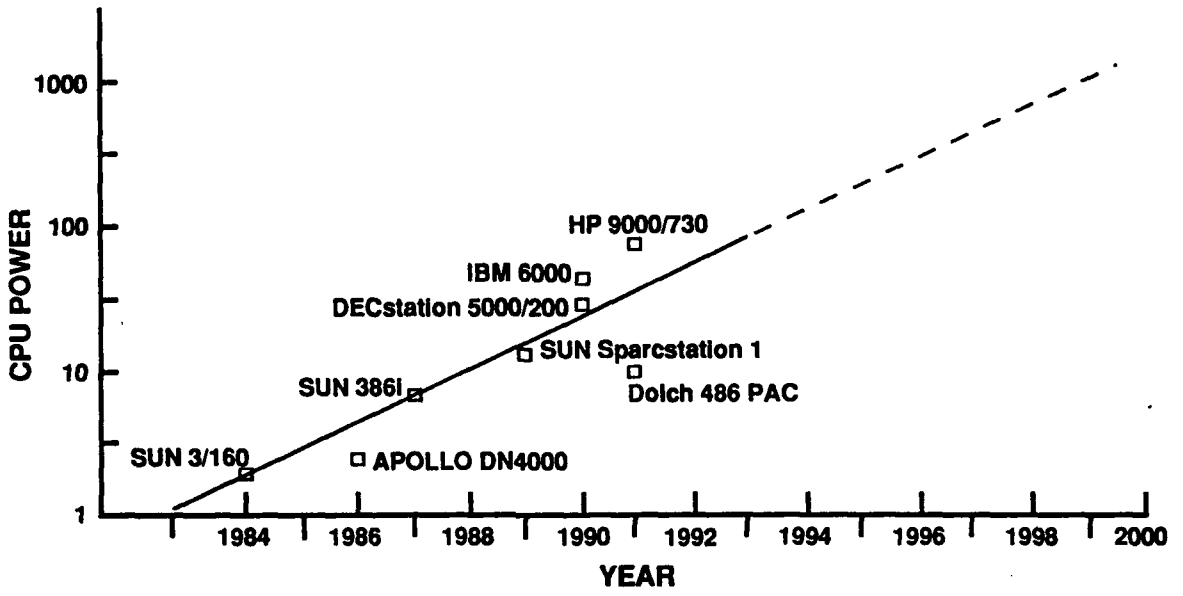


Figure 2. CPU Performance Trends for Workstations (Normalized to VAX 11/780)

Table 4. Capabilities of Portable Computers

	Full Laptop	Mono Suitcase	Color Suitcase
Hardware Cost (35% off list)	$11,000	$13,000	$20,000
Weight (lb) (includes transit case, printer)	30 +battery	35 +battery	38 +battery
Print Maps	Vector Maps Only	Vector Maps Only	Vector Maps Only
Interface to External Systems	Floppy, Modem, Serial, Network	Floppy, Modem, Serial, Network	Floppy, Modem, Serial, Network
Estimated Response Time for Route Evaluation (vs 2 sec)	35 sec	20 sec	6 sec
Displays			
Maps	Vector Only	Vector Only	Vector Only
Threat Depiction, ADRI Perspective View, Radar Prediction and EO/IR	Limited Greyscale, Slow Response	Limited Greyscale, Slow Response	Yes
Estimated Response Time for Radar Prediction (vs 12 sec)	190 sec	120 sec	28 sec
ULOSA Compliance	Yes	Yes	Yes
Secure OS	Yes	Yes	Yes
Key Limitations	Limited Disk Space No Expansion Slots No EOD	No EOD	-----

5. ULOSA STANDARDS

A Unit Level Open System Architecture (ULOSA) standard that takes advantage of trends in commercial developments has been developed to provide a standard for physical and electronic exchange of information between systems and to allow for the rehosting of software. The ULOSA standard was originally developed by The MITRE Corporation in support of ESD to provide an interoperability standard for Tactical Air Force unit level systems including mission planning, intelligence systems (e.g., Sentinel Byte) and operational systems (e.g., WCCS). Based on common commercial and military standards it was designed to take maximum advantage of existing, working, commercially available products. Its primary purpose is to define a standard to facilitate interoperability for unit level systems. It does not replace system-to-system interface control documents which stipulate the actual data format content to be transferred. Additionally it seeks to establish commonality for unit-level system environments (e.g., graphics processing, operating systems) and to further enhance portability and extensibility.

In the area of interoperability, the standard covers unit-level network protocols and services, extensions, and management; a transition plan to Government Open System Interconnect Profile (GOSIP); non-network intersystem communications; and security constraints. A key element was the preservation of all levels of data exchange mechanisms typically found in the field--floppy disk, synchronous and asynchronous serial, and network. This is critical when systems are rapidly deployed to unprepared facilities.

In the area of system environment the standard covers graphics processing, database query, operating system interfaces, windowing and data element standardization. In addition,

standards for interchange formats for word processing, spreadsheets, imaging, graphics are recommended. Table 5 provides a summary of the recommended standards.

These standards continue to evolve and, as later discussed in this paper, continue to have performance and portability implications. The choice of the "perfect" set of standards results in extensive controversy. An evolving, yet compatible set of standards is inevitably the best compromise. Nevertheless standardization is necessary if re-usability and portability are to be achieved. We have preferred approaches that invited compliance motivated by cost and schedule savings, or compliance that results by common usage, rather than mandating compliance.

The ULOSA standard will continue to evolve as future technologies and standards solidify: TCP/IP will evolve to GOSIP with the expected co-existence of TCP/IP and GOSIP protocol suites for some time (figure 3). In addition Ethernet will evolve to Fiber Distributed Data Interface (FDDI) with approximately 100 megabits/second capacity and ultra high speed LAN's (from one to six gigabits/seconds) if higher communication transfer rates are required.

Some key issues which remain are:

- ULOSA recommends OSF MOTIF for common "look and feel;" however, a clear standard has yet to emerge. ULOSA also recommends PHIGS for 2D/3D Graphics; however, GKS and proprietary graphics packages still prevail in the marketplace. We believe, based on recent research/product evaluations, that the PHIGS extension to X windows (PEX) is the likeliest candidate for a single graphics software standard to support mission planning (Ref. 2).

- The implications of the co-existence of such standards as Ada, POSIX-compliant operating systems and X-Windows has not been fully evaluated. In some cases the binders between these products and other standards are not yet commercially available, especially since "C" is more frequently used in the POSIX/Unix world than Ada.

- ULOSA does not specify a software architecture. The role of object oriented design and methods for ensuring software re-usability are still under investigation. Clearly, however, the software architecture is the key to the expandability/adaptability of the system (Ref. 6). For AFMSS, industry has been tasked to develop an aircraft specific interface to the common Air Mission Planner to define this architecture. This experience should prove very valuable in specifying software architectures for mission planners in the future.

6. GRAPHICS WORKSTATION PERFORMANCE

There are several types of graphics displays used in a typical mission planning operation:

1. Plan views of maps and images
2. Profile views of terrain and routes
3. Overlays of threats targets, routes, contours, zones, and weather
4. Projections of electro-optical and radar screens
5. Perspective views of imagery, features (DFAD), and obstructions (PVOD) over elevation (DTED)

These will be used in conjunction with a user-system interface consisting of the usual collection of desk top features, such as pull-down menus and scroll bars, along with movable and scalable windows.

In order to generate these graphics displays, the graphics services needed include raster operations (i.e., bit mapped displays), vector drawing of lines in two and three dimensions, drawing of polygons in two and three dimensions (Ref. 7), area fills, and drawing of text/symbols. The extent to which standards provide these services is shown in Table 6.

Transfer Formats	
Word Processing Files	ASCII, ODA
Spreadsheet Files	ASCII
Imagery Files	NITF
Graphics	CGM, IGES, or DXF
Weather	FCM-S2-1986

System Environment	
Windowing	X Window System
Data Element Standardization	JINTACCS
Database Query	SQL
Graphics Processing	PHIGS
Operating System Interface	POSIX

OSI Protocol Layers	LAN Standards	Point-to-Point Standards
Application File Transfer Mail Virtual Terminal Name Services	FTP SMTP Telnet Domain Name System	FTP over SLIP, KERMIT X.400 Telnet over SLIP none
Transport	TCP, UDP	
Network	IP, ICMP	SLIP
Data Link	802.2, SNAP, Ethernet V2.0	
Physical	802.3, Ethernet V2.0	RS-232, RS-449/422, MIL-STD-188-114

Table 5. ULOSA Standards

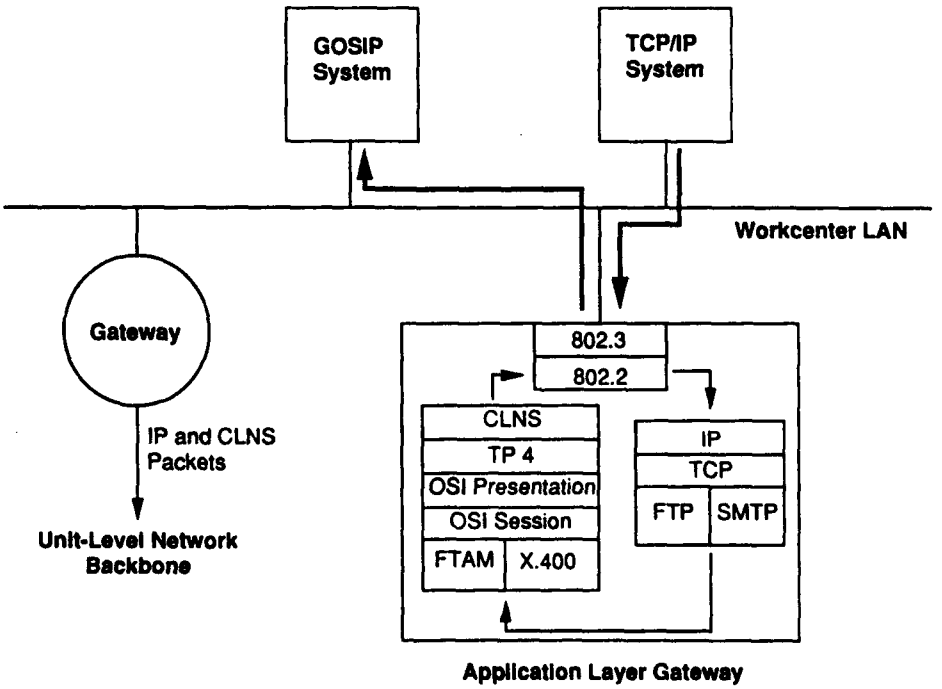


Figure 3. Application Protocol Translation

Table 6. Functional Capabilities of Graphics Standards for Mission Planning

Function	X Windows	PHIGS	PHIGS+	GKS
2D Drawing Functions:				
Raster Operations	Yes	No	No	No
Vector Draw	Yes	Yes	Yes	Yes
Text and Symbol Draw	Yes	Yes	Yes	Yes
Polygon Draw	Yes	Yes	Yes	Yes
Area Fill	Yes	Yes	Yes	Yes
3D Drawing Functions:				
Polygon Draw	No	Yes	Yes	No
Vector Draw	No	Yes	Yes	No
Text and Symbol Draw	No	Yes	Yes	No
Advanced Rendering: Smooth Shading; Lighting Models; Depth Cueing	No	No	Yes	No
Archival Storage	No	Yes	Yes	Yes
Editable Display List Storage	No	Yes	Yes	No
Support for Multiple Windows	Yes	No	No	No
Networked Operation	Yes	No	No	No

While X Windows handles displays of raster maps and image-based data such as satellite photography, it lacks functions to employ the workstations rendering hardware to generate geometric-based terrain visualizations, such as a drape of (DFAD) features over (DTED) elevations. On the other hand, the PHIGS+ standard can support highly realistic, geometry-based terrain visualizations handily with its 3D drawing primitives, but is incapable of dealing with raster images and, therefore with the display of digitized maps. GKS falls short in both of these areas.

There is very little data on response times of workstations utilizing standards in its server software packages. Our experience indicates that the response times in Table 3 can be met by workstations in the U.S. $25,000 to $50,000 range using their native graphics libraries. Some of these response times are already longer than ideal. The addition of standards such as X Windows and PHIGS adds additional performance penalties.

Some earlier studies conducted at MITRE (Refs. 8 and 9) indicate performance penalties ranging over factors from three to ten utilizing PHIGS or GKS instead of native graphics packages for 2D vector drawings and text/symbol drawings. Recent measurements of X Window performance indicate that severe (order of magnitude) performance penalties exist.

With none of the current standards meeting full functionality, and with the existence of major performance penalties, a truly portable, high performance software graphics standard solution is needed. That solution appears to be PHIGS extension to X (PEX). Not only does PEX provide all of the capabilities required for mission planning, but also it provides reasonable subsets for compatibility with different types of workstations, and it is being provided as the native graphics library by some vendors. It is the latter which gives the greatest encouragement by promising the kind of performance previously achieved only by proprietary graphics packages. This kind of standardization and performance will permit the portability, affordability and quality of display manipulations required by mission planning.

7. COMMON MAPPING SYSTEM

A mission planning system relies heavily on Mapping, Cartography, Geodesy and Imagery (MCG&I) data. Worldwide coverage of all types of digitized MCG&I data requires approximately two to three terrabytes of storage. Traditionally, paper charts have been digitized; and uncompressed, unmodified Defense Mapping Agency digital products

have been available organized by data type (e.g., Digital Terrain Elevation Data, Digital Feature Analysis Data, World Vector Shoreline). However, the aggregate volume of data becomes prohibitively large and unmanageable for field users, such as mission planners, whose area of operation are large and variable. (Typical MCG&I products and their storage requirements are provided in Table 7.) This has led to use of non-standardized, proprietary format, compressed data which has been costly to acquire, reproduce, store and distribute.

With hundreds of gigabytes of data required for planning on a workstation, there is a need to both index the information for rapid access and to develop compression/reduction techniques. Furthermore, there is a need to provide this data in a standardized, digitized format in the public domain and available with worldwide coverage. At the current disk drive cost of approximately U.S. $5,000/gigabyte, the data needs to be reduced/compressed to provide affordability and portability of the system. By providing regional storage and flexibility of access, terrabytes can be reduced to 200 gigabytes, and by further reduction/compression by factors of 48:1, the amount of on-line storage can be reduced to four to six gigabytes while maintaining the quality of the displayed/printed material. Such considerations have led the U.S. Air Force/ESD to develop a Common Mapping System (CMS) standard for MCG&I data.

CMS initially consists of a standardized, cartographic datebase structure. It will be augmented in the future with a set of standard software functions (toolkits) for exploiting the database, and a suite of tests for verifying that any given application has correctly implemented the database structure and the related functions. CMS uses the World Geodetic System (WGS-84) model of the earth's surface, to which all products are referenced. The database structure organizes data in 1° X 1° geocells for file access. Related data type can be stacked for matching geocells (Figure 4). It also provides for specifying the data compression/reduction method used.

By providing operationally selectable areas and appropriate compression/reduction of 48:1, four to six gigabytes of on-line storage can be used to provide the following coverage:

GLOBAL: Worldwide coverage containing WVS, ADRG GNC and JNC charts, and DAFIF

AREA: Theatre-wide coverage (equivalent to the coverage of one JNC chart) containing ADRG ONC and TPC charts, DCW data and DTED Level 1 (250,000 square nautical miles)

KERNEL: Kernel set of 70 user selected geocells (one degree by one degree) containing DTED and DFAD Level 1, ADRG JOG-Air, PVOD, and either ADRG TLM or ADRI.

An example of this particular configuration is depicted in example 1 (Figure 5). Additional configurations using the same amount of storage but different allocation of products are depicted in example 2 (Figure 6) and example 3 (Figure 7).

In performing investigations of compression/reduction techniques in the laboratory, MITRE has found that 48:1 and even 96:1 reduction compression ratios are feasible while maintaining both display and printer quality. Quality Judgment Criteria were background stipple and moire, legibility and aliasing of fine print differentiation of rivers and contour lines, preservation of symbology, and color fidelity. The work researched several spatial reduction techniques to achieve 4:1 spatial reduction. Of the filters investigated, neighborhood averaging gave the worst results and a separable cubic filter gave very good results.

The standard vector quantization (VQ) technique Linde Buzo Gray (LBG) was found to produce fairly poor results at 4:1 compression. However, other vector quantization techniques, especially "pairwise nearest neighbor," resulted in acceptable quality, provided color quantization (3:1) was well-integrated with VQ. Custom color table

Table 7a. MCG&I Products and Storage Requirements

MCG&I Product Sizes: Per-Product

Product	Data Type	Scale	Typical Coverage (lat x long @ 45^0 lat)	Original Paper Size (inches)	Total No. of Charts	Original Data Size (per chart)	Original Data Volume (per series)	Compressed Data Size (per chart)	Compressed Data Volume (per series)
ADRG GNC Series	Raster	1:5M	45^0 x 60^0	40 x 60	27	380 MB	10 GB	7.9 MB	0.21 GB
ADRG JNC Series	Raster	1:2M	16^0 x 32^0	40 x 60	122	380 MB	46 GB	7.9 MB	0.96 GB
ADRG ONC Series	Raster	1:1M	8^0 x 16^0	40 x 60	270	380 MB	100 GB	7.9 MB	2.1GB
ADRG TPC Series	Raster	1:500K	4^0 x 8^0	40 x 60	893	380 MB	340 GB	7.9 MB	7.1 GB
ADRG JOG-Air Series	Raster	1:250K	1^0 x 2^0	22 x 29	3,401	120 MB	400 GB	2.5 MB	8.5 GB
ADRG JOG-Grnd Series	Raster	1:250K	1^0 x 2^0	22 x 29	5,045	120 MB	590 GB	2.5 MB	13 GB
ADRG TLM Series	Raster	1:50K	12' x 20'	22 x 29	---	70 MB	---	1.5 MB	---
WVS	Vector	1:250K	Worldwide	n.a.	n.a.	n.a.	170 MB	n.a.	170 MB
DCW	Vector	1:1M	Worldwide	n.a.	n.a.	n.a.	10 GB	n.a.	10 GB
DAFIF	Text	n.a.	Worldwide	n.a.	n.a.	n.a.	---	n.a.	---

Table 7b. MCG&I Products and Storage Requirements (Concluded)

MCG&I Product Sizes: Per-Geocell

Product	Data Type	Scale	Compressed Data Volume
DTED Level 1	Gridded		2.95 MB
DTED Level 2	Gridded		27 MB
DFAD Level 1	Vector		0.4 MB
DFAD Level 1C	Vector		
DFAD Level 2	Vector		
DFAD Level 3C	Vector		
PVOD	Vector		0.5 MB
JOG-Air	Raster	1:250K	1.2 MB(1)
JOG-Grnd	Raster	1:250K	1.2 MB(1)
TLM	Raster	1:50K	29 MB(1)
ADRI	Raster	1:50K	29 MB(2)

Notes: 1) Compressed 48:1 from original DMA density
2) Compressed 4:1 from source data density (1 byte/pixel)

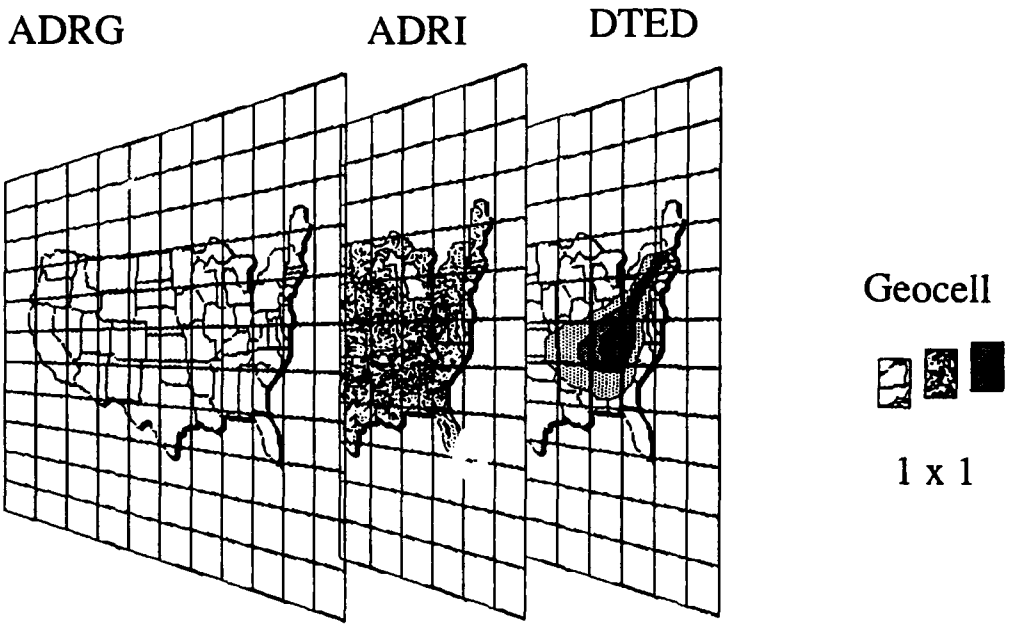


Figure 4. CMS Database

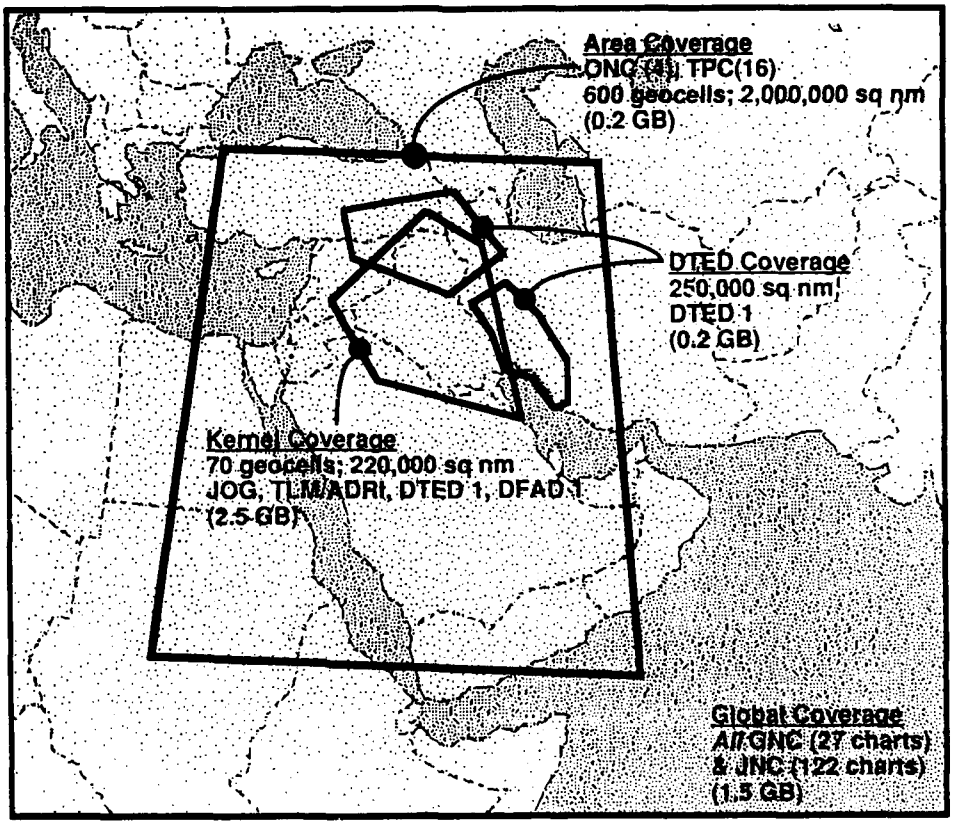


Figure 5. Example 1

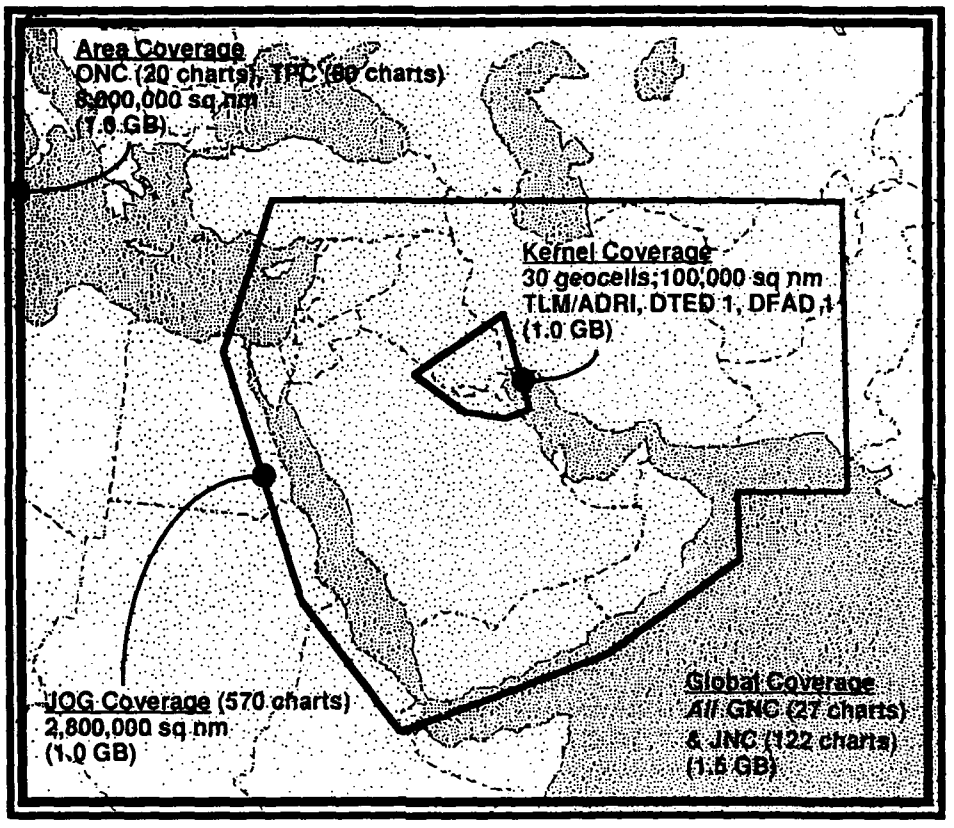


Figure 6. Example 2

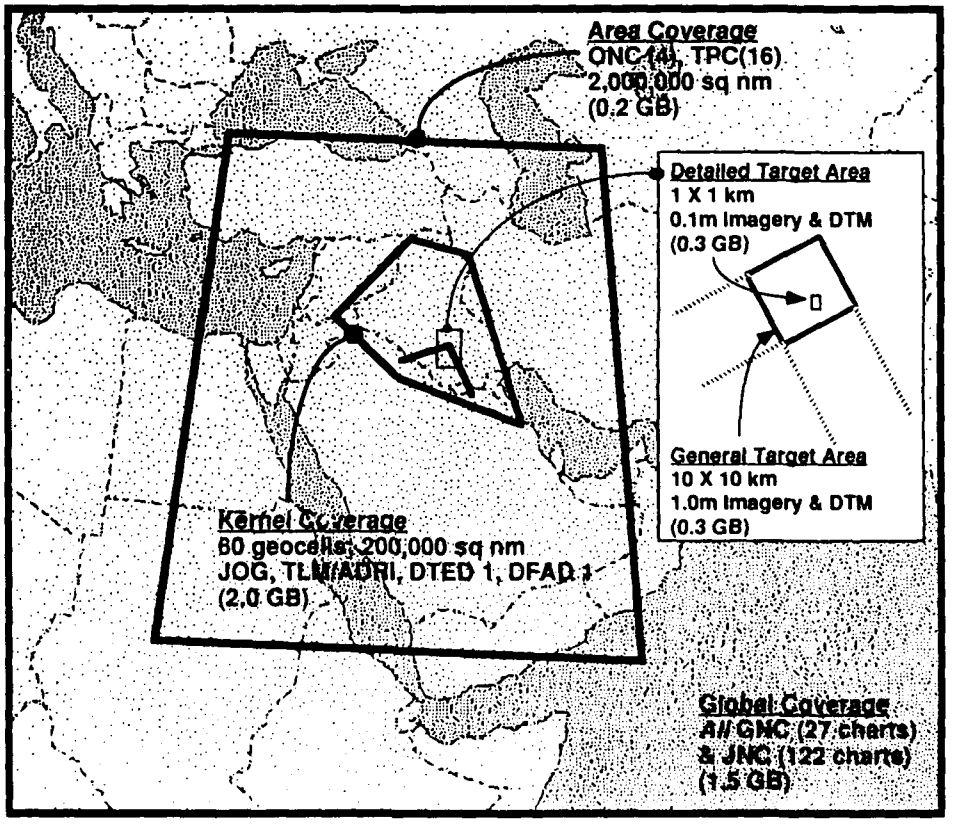


Figure 7. Example 3

techniques were needed to maintain compatibility with VQ compression schemes. Additional work is still required to specify standards for "quality" of displayed and printed maps.

Mission planning requirements for paper usable under high temperature/intense sunlight conditions, when night vision conditions, and with acceptable grey scale reproductions, color, contour line legibility, printer speed, and paper size still push the state of the art of printers if required to be available in a single printer. Media costs, which vary from less than U.S.$1.00 to $8.00 per page also play an important role in choosing an affordable mission planner.

Image data presents additional problems for two reasons: sheer quantity of data and the need for real-time transmission of some imagery data. Satellite image data has recently been found by the user to be a particularly useful planning tool. In addition to being used for terrain perspective views and fly-through, imagery is being used for targeting and battle damage assessment.

The large amount of storage and time required to transmit imagery is a function of the number of bits required to represent the image. There are similar techniques for the storage and transmission of data, text or imagery; however, the sheer volume of image data causes problems. While a standard page of text contains 25 thousand bits of data, an 8-bit pixel grey scale image (1024 x 1024 pixels) contains over 8 million bits and an equivalent size 24 bit/pixel color image over 25 million bits. Typical transmission times for text (one page) and imagery (1024 x 1024 pixels) are given in Table 8.

These numbers represent transmission times for uncompressed data. Algorithms yielding compression ratios of six to ten, without significant loss of legibility, currently exist; however, they can have significant cost and processing penalties.

8. FUTURE CHALLENGES

Some of the near-term challenges have already been indicated in this paper. These include the development of a software architecture that supports the addition of new aircraft and updates of avionics software; meeting performance goals with standard graphics packages, and providing useful mission planning capabilities on a "portable" mission planner. Others include moving map displays, standardizing on threat models, evaluating and testing automated routing aids, ensuring the availability of required data, such as weather data, and maintaining timeliness and integrity of huge databases. The goal is to provide an interoperable mission planner that can be used by any aircraft assigned to a base.

Future technology challenges include incorporation of Artificial Intelligence techniques, including "learning" from previous missions, interfacing with mission rehearsal systems, in flight route/target updates, and taking advantage of technology breakthroughs in such areas as natural language interfaces and virtual reality.

Table 8. Transmission Times (In Minutes)

Data Rate	Text (One Page)	Greyscale (1024 x 1024 pixels)	Color (1024 x 1024 pixels)
Commercial Channel			
56000 bits/sec	.01	2.5	7.5
Tactical Channel			
16000 bits/sec	.03	8.7	26.2
Telephone or FAX			
9600 bits/sec	.04	14.6	43.7
1200 bits/sec	.35	116.5	349.5

9. REFERENCES

1. DeRosa, J. K., "Technical Challenges in Mission Planning," Automated Mission Planning Magazine, A Publication of the Aerospace Education Foundation--NJ, Spring 1990, pp. 16-17, and 37.

2. Conway, J. L., "Standard Graphic Environments for Mission Planning," Proceedings of the Mission Planning Symposium, held at The MITRE Corporation, Bedford, Massachusetts, published by the Armed Forces Communications and Electronics Association (AFCEA)/Lexington-Concord Chapter, 24-25 September 1991.

3. Brady, J. T., "A Theory of Productivity in the Creative Process," IEEE Computer Graphics and Applications, May 1986, pp. 25-34.

4. Breen, P. T., "An Overview of Workstation Technology," Society for Information Display Conference Tutorial Course, 1989.

5. Santo, B. and Wolland, K., Associate Editors, "The World of Silicon: It's Dog Eat Dog," IEEE SPECTRUM, Volume 25, Number 9, September 1988, pp. 30-39.

6. Horowitz, B. M., "The Importance of Architecture in DOD Software," The MITRE Corporation, Bedford, MA, Report No. M91-35, July 1991.

7. Southard, D. A., "Piecewise Planar Surface Models from Sampled Data," Scientific Visualization of Physical Phenomena, N. M. Patrikalakis, Editor, Springer-Verlag, Tokyo, 1991, pp. 667-680.

8. Conway, J. L.; Freiter, S. J.; and Leger, J. R.; "Super Graphics Workstations for Real-Time C3 Applications," Proceedings of the Society for Information Display, Vol. 32/1, 1991, pp. 49-53.

9. Conway, J. L. and Leger, J. R., "Graphics Benchmarks for Real-time C3 Applications," Proceedings of the Society for Information Display, Vol. 30, No. 1, 1989.

10. ACKNOWLEDGEMENTS

This work was supported by the Electronic Systems Division of the U.S. Air Force Systems Command, Hanscom Air Force Base, Massachusetts.

The MITRE Corporation Mission Planning Project provides System Engineering support to the Mission Planning Program Office under the direction of Lt. Colonel A. P. Sharon. The results compiled in this paper were in large part the work of the MITRE mission planning project staff.

Search Route Planning

J. R. Van Zandt
The MITRE Corporation
Burlington Road
Bedford, MA 01730
Mail Stop T120
USA

1. SUMMARY

Planning an airborne search for a relocatable target involves minimizing the risk to the crew while maximizing the estimated likelihood of finding targets. When the number of potential sites is small, the problem reduces to finding the best (usually the shortest) route connecting them. The aircraft need not overfly each site, but must only come within sensor range of it. Thus, the search route planning problem can be modeled as a *covering salesman problem* (CSP) -- a generalization of the traveling salesman problem in which the tour need not visit every city, but every city must be within a certain distance of some city on the tour. The spacefilling curve heuristic for the traveling salesman problem is demonstrated, then a new heuristic for the covering salesman problem is presented and applied to the search route planning problem. When the number of sites is large, the planner must also decide which subset of them can be visited. A genetic algorithm has been developed for selecting both a set of sites to visit and an appropriate routing pattern. The procedure also accounts for the finite turning radius of the aircraft.

1.1 Characteristic Distances

First, it may help to systematically introduce the constraints on the search routing problem. Several important aspects of the problem may be described by a set of *characteristic distances*:

R_{turn}, the minimum turning radius,
R_{sensor}, the sensor range,
R_{site}, the typical intersite distance,
R_{area}, the diameter of the search area, and
R_{flight}, the flight distance permitted by time and fuel constraints

We may expect at least

$$R_{turn} << R_{area} < R_{flight} \quad (1)$$

so the aircraft can fly across the search area and maneuver within it,

$$R_{sensor} << R_{area} \quad (2)$$

so the sensor can cover only a small part of the search area at once, and

$$R_{site} << R_{area} \quad (3)$$

so the search area contains several sites. The simplest situation is

$$R_{turn} << R_{sensor} << R_{site} << R_{area} << R_{flight} \quad (4)$$

This can be modeled as a planar traveling salesman problem, which is efficiently solved by the heuristics of section 2.

However, the aircraft need not overfly each site, but must only come within sensor range of it. An overflight of one site may provide adequate sensor coverage of nearby sites as well. This becomes significant if intersite distances can be comparable to the effective sensor range, giving us the partial ordering

$$R_{turn} << R_{sensor} \sim R_{site} << R_{area} << R_{flight} \quad (5)$$

This can be modeled as a planar covering salesman problem, which is also discussed in section 2 and appendix A. In this case, the search route planning problem can be modeled as a *covering salesman problem* (CSP) as formulated by Current and Schilling [1]. The CSP is a generalization of the traveling salesman problem in which the tour need not visit every city, but every city must be within a certain distance of some city on the tour.

Assume there are n sites. If the search area were square and the visited sites were distributed uniformly, the length of the shortest tour would be roughly $\sqrt{n}\ R_{area}$ (see appendix A). A fuel or flight time constraint becomes significant only when it means that not all sites can be covered. In terms of these characteristic distances, this corresponds to the case

$$R_{area} < R_{flight} < \sqrt{n}\ R_{area} \quad (6)$$

The assumption of small turning radius is appropriate for high altitude searching, because of the necessarily large sensor range in that case. It is not valid for low altitude searching. The general low altitude case is then

$$R_{turn} \sim R_{sensor} \sim R_{site} << R_{area} < R_{flight} < \sqrt{n}\ R_{area} \quad (7)$$

1.2 Overview

Section 2 discusses the CSP, and heuristics for their approximate solution. Section 3 shows how a flight path can be constructed which accounts for the finite turning radius of the aircraft.

When the number of potential sites is large, the aircraft's time and fuel constraints may preclude visiting all of them. The planner must then decide which of them are to be visited. Section 4 extends the above procedure to include the selection of an appropriate set of sites. The new procedure also evaluates alternative routing patterns and accounts for the finite turning radius of the aircraft.

Selection of the sites to visit is handled by imbedding the heuristic router within an optimization program. The outer program selects a set of sites to be visited. The heuristic

router finds a search route covering those sites. The outer program computes a score for the route. It then adjusts the sites to be visited in an attempt to optimize the score (while satisfying time and fuel constraints) and repeats. For this combinatorial optimization problem, a *genetic algorithm* is used.

Section 5 displays typical results. A concluding discussion appears in section 6.

2. ROUTING PROBLEMS

2.1 The Traveling Salesman Problem

In the traveling salesman problem (TSP), one is given a set of cities and a table of intercity distances, and is required to find a tour which visits all the cities while minimizing the total distance traveled.

In the Euclidean TSP, the graph is complete (so that travel is permitted between any two cities) and the intercity distances are calculated from the city locations using the Euclidean metric. We will be concerned with air travel among sites in a small region, which is well modeled by the planar (two-dimensional) Euclidean TSP.

The TSP is NP-hard, as discussed by Garey and Johnson [2]. This means that the time required to find the exact solution to a problem with n cities increases faster than any fixed power of n, and that for practical purposes, large problems (thousands of cities) cannot be solved exactly. However, there are a number of heuristics for the TSP which give good solutions (i.e., within a few percent of the length of the optimal solution).

2.1.1 *The Spacefilling Curve Heuristic for the TSP*

Bartholdi and Platzman have found a heuristic for the planar TSP [3] using a spacefilling curve due to Sierpinski, which is a continuous map ϕ from the unit interval $C = [0,1]$ onto the unit square $S = [0,1]^2$. Sierpinski's curve is the limit of the sequence shown in figure 1. It is a closed curve in which 0 and 1 are mapped to the lower left corner.

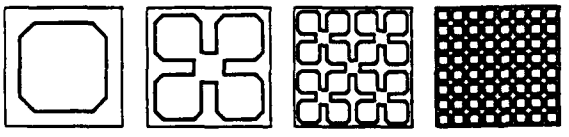


Figure 1. Approximations to Sierpinski's Spacefilling Curve

Their heuristic solution to the planar TSP is as follows:

HEURISTIC TSP-1

a. *For each point $p \in [0,1]^2$ to be visited, find a key θ such that $p = \phi(\theta)$.*
b. *Sort the points by their corresponding θs.*
c. *Visit the points in order of increasing θ, then return to the first point.*

The feasibility of this procedure rests on an efficient algorithm for computing the θ values, or *keys*. Given n points, the keys may be computed and sorted in $O(n \log n)$ time or, using BINSORT, in $O(n)$ expected time.

The method is illustrated in figure 2, where the city locations have been chosen to lie on the curve at the second level of refinement. The resulting tour visits the cities in the same order as the spacefilling curve.

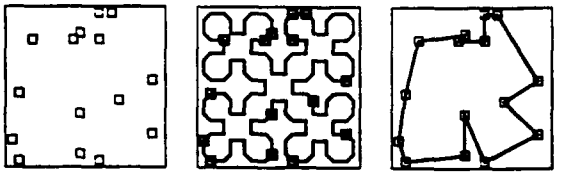


Figure 2. Using a Spacefilling Curve to Find a Tour

The same method can be extended to Euclidean traveling salesman problems in higher dimensions, using higher dimensional spacefilling curves [4].

This heuristic produces efficient tours because a spacefilling curve preserves nearness: points close together in C map (via ϕ) onto points close together in S. In S we use the Euclidean measure denoted by $|p_i - p_j|$. Sierpinski's curve is a circuit with $\phi(0) = \phi(1)$, so the natural metric on C is $D(\theta,\theta') \equiv \min\{|\theta-\theta'|,1-|\theta-\theta'|\}$. Its nearness property may be stated formally as [3]:

$$R(\theta,\theta') \equiv \frac{|\phi(\theta)-\phi(\theta')|}{2\sqrt{D(\theta,\theta')}} \leq 1 \qquad (8)$$

Since C is one dimensional and S is two dimensional, it should perhaps not surprise us that a distance in S tends to be proportional to the square root of the corresponding distance in C. (For a spacefilling curve mapping C to k-space, a distance in k-space tends to be proportional to the kth root of the corresponding distance in C.) Note that there is no corresponding bound on the inverse of R, so while nearness along C guarantees nearness in S, the converse does not hold. That is, points nearby in S are only *likely* to correspond to points nearby in C.

2.1.2 *Tour Improvement*

For TSPs in which the cities are distributed uniformly on the unit square, TSP-1 generates tours which are about 25 percent longer than the expected length of the optimal tour [3]. These heuristic tours often have fairly obvious inefficiencies.

Platzman and Bartholdi [4] have also described an improvement procedure which perturbs the location of each point just enough to change its position in the tour and keeps the changes that shorten the tour. On average, the enhanced heuristic gives tours about 15 percent longer than optimal, and still takes $O(n \log n)$ time. We will refer to it as TSP-2.

TSP-2 was used to find a tour through 100 cities placed randomly in a square. The resulting tour is shown in figure

3. Note the resemblance between the tour and the spacefilling curve.

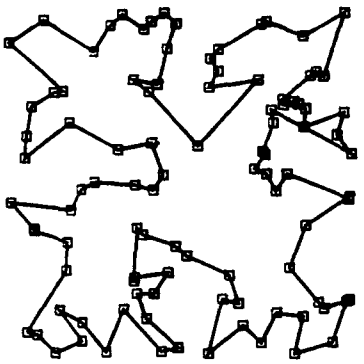


Figure 3. Example of a Heuristic Tour through 100 Cities in a Square

2.1.3 *Open Tours*

A search route for an aircraft should begin and end at the edge of the search region, but need not begin and end at the same point. In fact, it is usually best if the beginning and ending points are widely separated. This can be arranged by using an open spacefilling curve (i.e., one that does not begin and end at the same point), rather than the Sierpinski curve.

One such curve is Hilbert's curve [5], shown in figure 4. To illustrate the use of this curve, TSP-2 was used with the Hilbert curve to find a tour through the same 100 cities as in the previous example. The resulting tour is shown in figure 5. Note that the start and end points are now in adjacent corners.

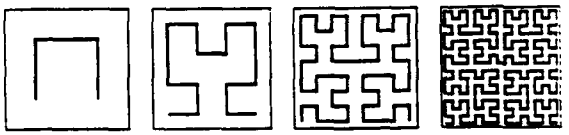


Figure 4. Approximations to Hilbert's Spacefilling Curve

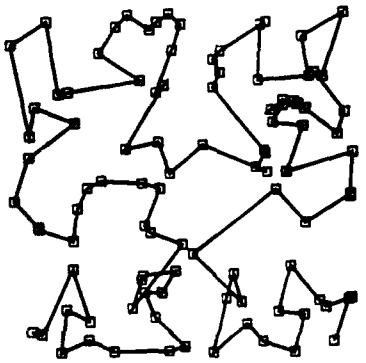


Figure 5. Example of a Heuristic Tour Generated Using the Hilbert Curve

2.1.4 *Other Heuristics for the TSP*

The "strips" method described by Karp and Steele [6] is an older heuristic for the planar TSP. For n cities, it consists of the following:

STRIPS HEURISTIC

a. Divide the area into $\sqrt{n}$ horizontal strips.
b. Sort the cities along each strip.
c. Visit the cities in the first strip from right to left, then those in the second strip from left to right, etc.

The "strips" heuristic can be implemented in the same way as the spacefilling curve heuristics, by defining an appropriate ordering function, so it also requires only $O(n \ln n)$ steps. For large numbers of sites distributed uniformly in the unit square, the strips heuristic generates TSP solutions about 22 percent longer than optimal (*vs.* 25 percent over optimal for the spacefilling curve heuristic using Sierpinski's curve). The strips heuristic is more sensitive to local variations in site density.

The "arbitrary insertion" heuristic for the TSP described by Golden [7] has also been adapted to the search routing problem as follows:

ARBITRARY INSERTION HEURISTIC

a. Start with a path directly from the start to the finish.
b. Repeat until all cities have been added:
1. Arbitrarily select city k not in the path to enter the path.
2. Find the edge (i,j) in the path which minimizes the cost of inserting city k and insert it there.

The arbitrary insertion heuristic requires $O(n^2)$ steps. It finds TSP solutions about 5 percent larger than optimal.

The arbitrary insertion method could be used in a semiautomated router in which an operator specifies part of the route (initial and final points, and some of the intermediate sites) and the computer adds the remaining sites. The initial route could include multiple visits to some sites.

The spacefilling and strips heuristics require a preprocessing step: the city locations must be rotated and scaled into the unit square. This is not necessary for the arbitrary insertion heuristic.

2.2 The Covering Salesman Problem

The Covering Salesman Problem (CSP) was formulated by Current and Schilling [1]. It is a generalization of the traveling salesman problem in which the tour need not visit every city, but every city must be within a certain distance d of some city on the tour. For example, suppose one is planning routes for a rural health care delivery team. It is sufficient that every potential patient be within some predetermined travel distance of a stop on the route. The CSP reduces to the TSP if d is sufficiently small.

In our application, the aircraft must pass within sensor range of each potential target site. We will approximate this by routing the aircraft directly over some subset of the sites, such that every site not visited is within sensor range of some visited site.

2.2.1 *The* COVTOUR *Heuristic*

Current and Schilling presented a heuristic for the CSP which they call COVTOUR [1]:

HEURISTIC COVTOUR

a. *Identify the minimum subset R of points which cover all the points to be visited. This is a set covering problem (SCP).*
b. *Find the minimum tour connecting the points in R. This is a TSP.*
c. *If more than one optimal solution to the SCP was found in step a., repeat step b. for each. The shortest tour found is the solution.*

Since both the set covering problem and the TSP are NP-hard as discussed by Garey and Johnson [2], this procedure is impractical for large n.

2.2.2 *Fast Heuristics*

Clearly, one naive heuristic for the CSP is to neglect the covering distance and treat it as a TSP, thereby finding a tour that visits every city. When the CSP is Euclidean, a spacefilling curve heuristic can be devised which is more effective: find a heuristic tour through all the cities, then choose a *subset* of cities that will cover all the cities. (Note that solving the routing problem first and the covering problem second is just the opposite of COVTOUR.)

This method for solving the CSP is actually an application of the following general technique proposed by Bartholdi and Platzman for solving combinatorial problems in the plane [8]:

GENERIC SPACEFILLING HEURISTIC (GSFH)

a. *Transform the problem in the unit square, via a spacefilling curve, to a problem on the unit interval.*
b. *Solve the (easier) problem on the unit interval.*

A family of heuristics of this type for a specific problem can be generated by using different transformations to the unit square, different spacefilling curves, and different solutions of the problem on the unit interval.

The spacefilling curve heuristic for the TSP is another special case of the GSFH.

We have developed several such heuristics, with different methods of selecting the subset of cities to be visited. These are discussed in detail in appendix A. The most effective heuristic is the one we have called CSP-4, and it is the one used for the examples in the remainder of this report. The solution found by CSP-4 for a sample problem with 21 cities is shown in figure 6. The circles indicate the coverage of the visited cities.

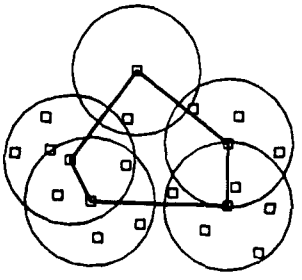


Figure 6. A CSP Solution Found by Heuristic CSP-4

2.3 Search Routes

Efficient search routes can be generated by combining the ideas of the previous two subsections with a simple way to transform the candidate target sites into the unit square.

In our application, we can assume the sites to be searched will be contained within a rectangle, and the search route should begin along one edge of the rectangle and end along the opposite edge. The spacefilling curve heuristic can generate such a route using the Hilbert curve if the rectangle is suitably transformed into the unit square. The transformation consists of a rotation, a displacement, and uniform scaling. (Nonuniform scaling is generally not helpful.) The final orientation and spacefilling curve should be chosen so that the search start and end points are near the desired ones.

3. FLIGHT PATHS

A flight plan includes a set of *waypoints* which a plane flies through in succession. In this work, waypoints are two dimensional. When the waypoints are well separated, the flight path can be approximated by straight lines between the waypoints. However, for closely spaced waypoints, high aircraft speeds, and/or restricted aircraft maneuverability, the finite turn radius of the aircraft must be taken into account.

By convention, a flight path consists of straight lines and circular arcs. One planning objective might be to find the shortest such path which passed through the specified sites. It might be constructed using the following manual procedure: Start with a flat map of the flight area. Insert a pin at each of the points to be visited. Drop over each pin a hollow cylinder of a size corresponding to the desired turn circle. Thread a string around the cylinders in the chosen order and draw it tight, forcing each cylinder against its pin. (We assume the cylinders do not interfere with each other.) The string then follows the shortest path for the chosen visitation order.

This manual procedure generates a flight path which visits each waypoint halfway through a turn. This doubles the number of significant locations the navigator must keep track of (search sites plus turn points). It requires accurate flying, since an incorrect airspeed or bank angle will lead to an incorrect turn radius and possibly a missed search site. It also means that the aircraft will never be level when overflying the site. For these reasons, as well as convenience, we have instead implemented the simpler *waypoint turn algorithm*: The aircraft initiates each turn as it passes a waypoint. The aircraft then flies along a *turn circle*, with its center at a distance R from the the waypoint and at right angles from the original flight path. Ordinarily, the aircraft turns toward the next waypoint, as shown in figure 7. However, if this would put the next waypoint within the turn circle, the turn can be made in the opposite direction instead, thus adding a *loop*. This is shown in figure 8. These two cases suffice to handle arbitrarily difficult routing problems, such as the closely spaced waypoints shown in figure 9.

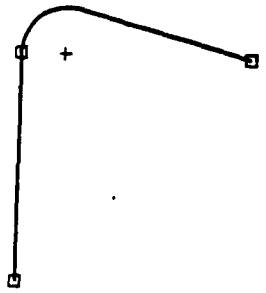


Figure 7. Aircraft Turns Toward Next Waypoint

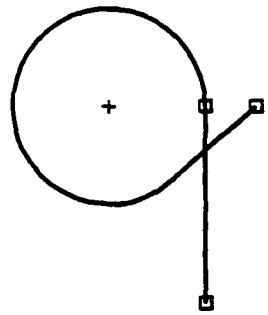


Figure 8. Aircraft Turns Away from Next Waypoint

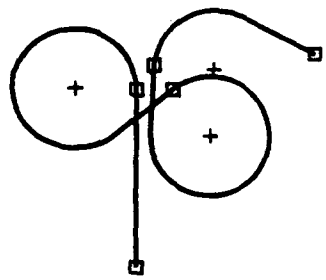


Figure 9. Waypoint Turn Algorithm Applied to Closely Spaced Waypoints

4. GENETIC ALGORITHMS

The traditional optimization procedures are calculus-based methods (solve for locations where the gradient vanishes, or *hill climbing*: move in the direction of the local gradient), *random* searches (generate points in parameter space at random and keep the best), and *enumeration* (generate all possible points and keep the best). Calculus-based methods can be applied only to differentiable objective functions, can be unstable, and are susceptible to the *foothill problem* -- getting trapped at a local optimum. The random and enumerative methods can be hopelessly inefficient when the search space is large.

A genetic algorithm (GA) is an alternative optimization procedure inspired by Darwinian evolution. A conventional GA uses a bit string which encodes the problem parameters. The GA keeps a population of ~*100* such strings and applies genetic operators to them to generate new strings. Each new string is evaluated, and better strings are preferentially retained for the next generation.

GAs are more generally applicable and robust than calculus-based methods, and can be much more efficient than random search and enumerative methods. GAs have been successfully applied to pipeline control [9], structural optimization [10], recursive adaptive filter design [11], VLSI circuit layout [12], and many other problems.

GAs were developed by John Holland and his colleagues at the University of Michigan [13]. A short description of GAs appears in the next two subsections. A more extensive discussion may be found in [14].

4.1 Description

The genetic algorithms in this work follow this reproductive plan:

GENETIC ALGORITHM

a. Create a random population of size N.
b. Sort the population in ascending order of fitness.
c. Repeat until there are N + m members in all:
  1. Choose randomly a member C_i, where $1 \le i < N$ (uniform distribution). C_i is almost unbiased.
  2. Choose randomly a member C_j, where $i < j \le N$ (uniform distribution). C_j is biased toward more fit members.
  3. Choose randomly a genetic operator g_k (see below).
  4. Create a new member $C \leftarrow g_k(C_i,C_j)$ or $C \leftarrow g_k(C_j)$ if g_k uses only one member. Discard C if it is the same as either parent.
  5. Determine fitness of C.
d. Merge the new members into the population, discarding duplicates.
e. Remove all but the last (strongest) N members of the population.
f. Terminate or go to step c.

m was set to *.20N* to reduce the sorting effort without materially delaying entry of new members into the active population. However, in a typical application, determining the fitness of new members dominates the computational effort.

The most important genetic operator is *crossover*, shown in figure 10. This operator chooses two positions along the bit string and uses the enclosed fragment from one parent and the end fragments from the other parent. Less important is *mutation*, which reverses one or more bits chosen at random from the single parent, shown in figure 11.

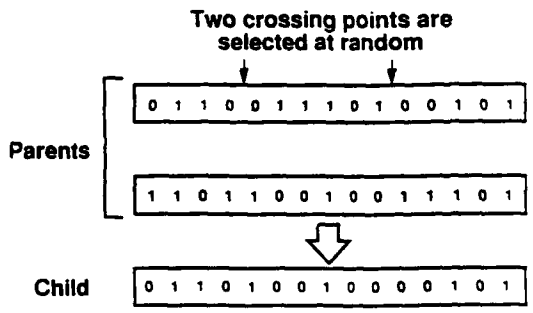


Figure 10. Crossover Operator

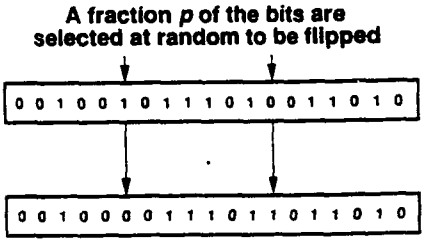


Figure 11. Mutation Operator

The probabilities with which different genetic operators are chosen are themselves adjusted during the computation. Details are discussed in appendix B.

4.2 Example

Suppose the problem is to find the value of x which maximizes

$$f_{12}(x) = e^{-4\ln 2\left(\frac{x-0.0667}{0.8}\right)^2} \sin^6(5.1\pi x + 0.5)$$

for the range $0 \le x \le 1$, a problem suggested by Goldberg and Deb [15].

In this case, the bit string has a single field of 16 bits. Let these bits be the Gray code for the integer i. (The Gray code is used instead of the usual binary code so that codes for neighboring values will differ by only one bit.) x is then simply $i/65536$.

Crossover and mutation are the only genetic operators used.

An initial random population of 60 members is shown in figure 12. By the 10th generation, shown in figure 13, members are clustering around the local peaks of the fitness function. By the 30th generation, shown in figure 14, all members are clustered around the highest peak.

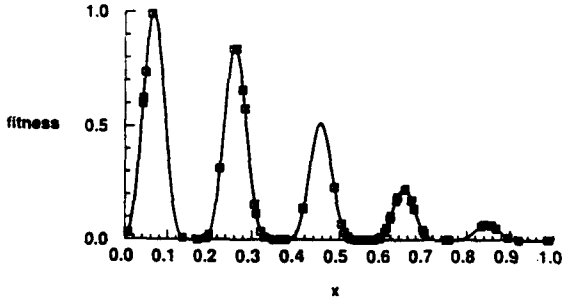


Figure 12. Initial Population

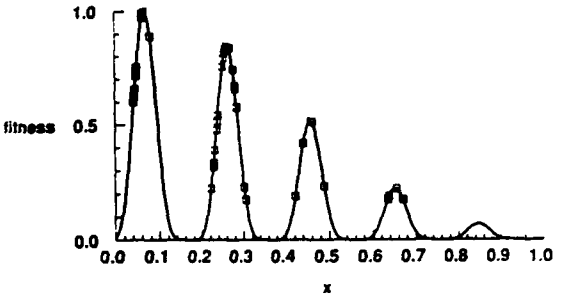


Figure 13. 10th Generation

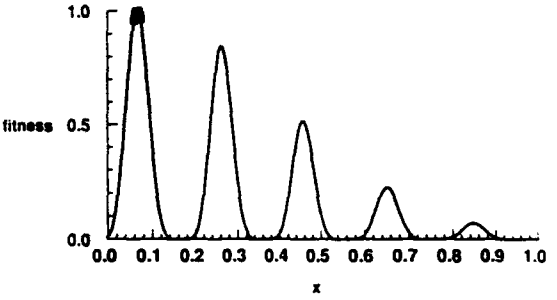


Figure 14. 30th Generation

4.3 Application to Routing

For the routing problem, a 4-bit field is used to select among 16 routing patterns that governed the heuristic solution of the traveling salesman problem. The patterns comprise the Sierpinski curve, the Hilbert curve, and 14 versions of the strips heuristic with different numbers and orientations of strips.

The remainder of the bit string consists of one bit per potential site, which is set if the site is to be visited and zero otherwise.

The "raw" fitness function is the sum of the values of the visited sites. The constraint on flight path length is handled by adding a *penalty function* as suggested by Goldberg [14]. Members satisfying the constraint are said to be *feasible*. Details are discussed in appendix C.

Two genetic operators are used in addition to the crossover and mutation operators described above.

The *mutate-routing* operator, shown in figure 15, replaces the routing pattern with another chosen at random. The primary motivation for this is that changing the routing pattern and changing the visited sites are significantly different operations, perhaps best applied at different rates. Defining separate mutation operators allows the operator probabilities to adapt independently (see appendix B). Early in the calculation we would expect that changes in the routing pattern might well reduce the search route length, so that the probability of selecting the *mutate-routing* operator should be moderate. Late in the calculation we would expect that for a given member, the routing pattern and the set of sites to be visited would be tuned to one another. A change in routing pattern would be unlikely to decrease the search route length. However, site changes might still improve the search (by removing a site to shorten the path length or adding a site to increase the

weight). Therefore it would be reasonable to mutate sites more often than routing patterns.

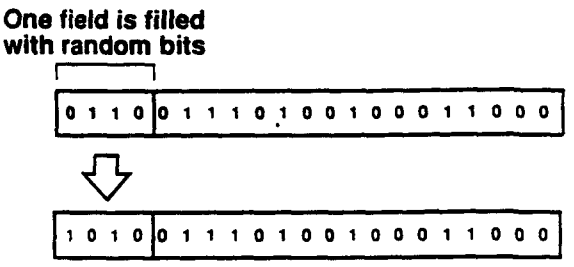


Figure 15. Mutate-Routing Operator

The other operator was motivated by the observation that the algorithm frequently chose routes that visited relatively low weight sites while skipping nearby sites with higher weights. This appeared to be due to the turn radius limitation. Merely adding the higher weight site may lead to the addition of a loop, whose length makes the path infeasible. On the other hand, deleting the low weight site reduces the value of the path. Thus moving the visit to the higher weight site would require two mutations (deleting one site and adding the other) but the intermediate states have low value and would be unlikely to survive. In GA terminology, the problem is *deceptive*.

The deception is overcome by combining the two mutations to form the *move-bit* operator. This operator, shown in figure 16, chooses a visited site at random and clears the corresponding bit in the bit string. It then finds the preceding and following set bit in the string and sets some different bit in the interval between them. This procedure makes sense only because the sites are rearranged in a preprocessing step so that neighboring bits represent nearby sites. (Actually, the sites are put into Sierpinski curve order.)

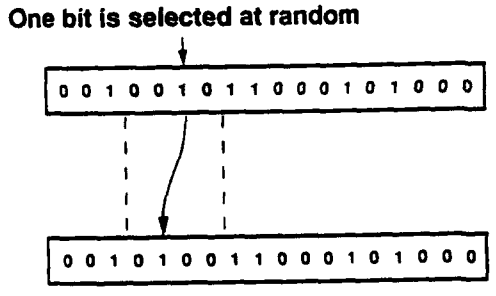


Figure 16. Move-Bit Operator

The progress of the calculation may be shown by plotting the raw fitness (sum of values of visited sites) of the best feasible member in each generation. Typical plots are shown in figure 17. The fitness increases rapidly at first, then more slowly. (The algorithm makes about 80 percent of its progress toward the final value in the first 20 percent of the time - a value which doesn't change much with different run times.)

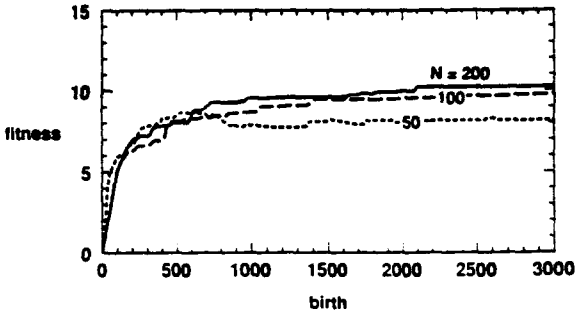


Figure 17. Score of Best Member

One of the important tuning parameters is the population size. Populations of up to several hundred members were used. Generally, large populations take longer to converge but reach better solutions. However, results are also poorer for very large populations when the total number of evaluations is held fixed. This is due to the tradeoff between population variability (i.e., presence of the building blocks for good solutions) and the number of generations.

If the population is too small, the members won't adequately span the search space. If the population is too large, then (for a fixed number of evaluations) not enough generations will pass to allow the best features of the various members to be combined. This is shown by the results for a 20 site problem shown in figure 18. Four runs were made for each population size, with each limited to 4000 evaluations. The figure shows the mean score and a one standard deviation range. The optimum population in this case is apparently near 150 members. The results for a 50 site problem shown in figure 19 suggest that the optimum population may increase for larger problems.

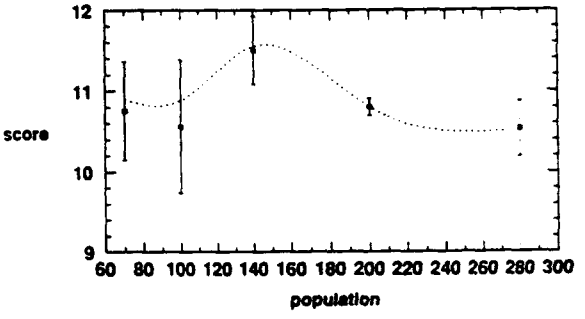


Figure 18. Scores as a Function of Population for 20 Site Problem

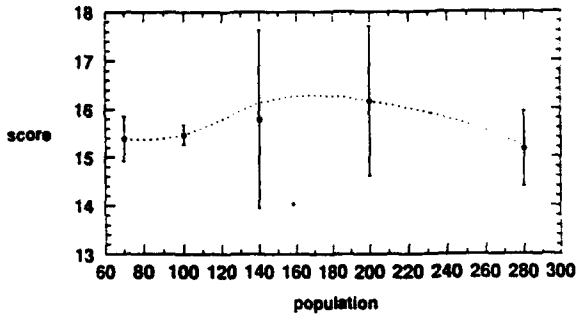


Figure 19. Scores as a Function of Population for 50 Site Problem

5. RESULTS

Figure 20 shows a sample problem with 20 sites. The path is required to start at the point labeled 1 and end at the point labeled 2. The other sites are shown as circles with areas proportional to their weight (that is, the expected likelihood of finding the target there).

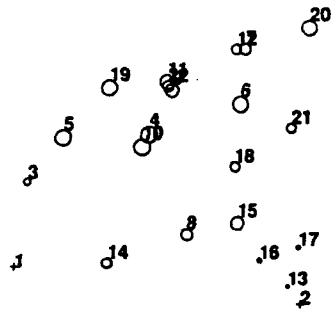


Figure 20. Sample Problem with 20 Sites

Figure 21 shows the best path found through those sites with path length restricted to 3.

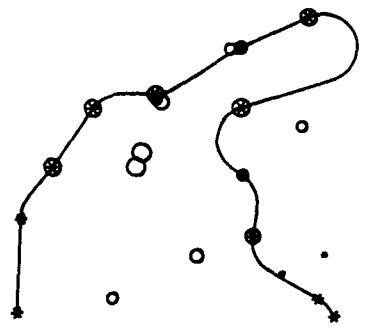


Figure 21. Sample Path Through 12 of 20 Sites

Figure 22 shows a second sample problem with 50 sites. The variation among paths found by the genetic algorithm can be judged from figures 23-25, which show the three best paths with length less than 5.

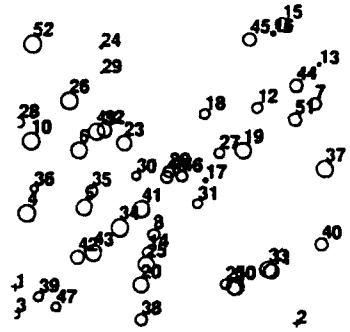


Figure 22. Sample Problem with 50 Sites

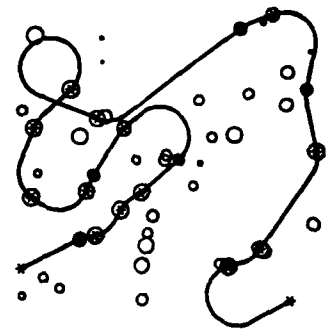


Figure 23. Path Through 20 of 50 Sites (Length 4.91, Weight 27.23)

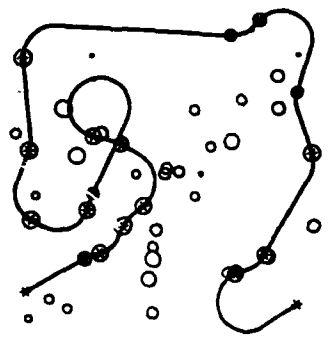


Figure 24. Path Through 19 of 50 Sites (Length 4.94, Weight 26.55)

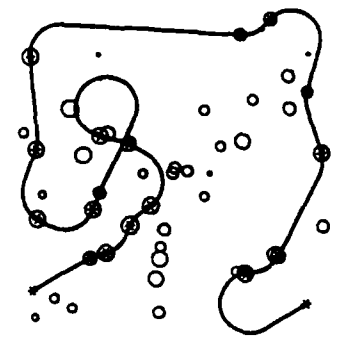


Figure 25. Path Through 19 of 50 Sites (Length 4.99, Weight 26.31)

None of the above runs used the arbitrary insertion routing procedure, relying instead on spacefilling curves and strips routing patterns. Detailed results and genetic algorithm parameters are listed in table 1.

Table 1. Parameters for Sample Results

Cities Visited	Weight	Length	Population	Eval.	Figure
20	12	11.14	2.78	120	4000	21
50	20	27.23	4.91	200	3000	23
50	19	26.56	4.94	200	3000	24
50	19	26.31	4.99	200	3000	25

6. DISCUSSION

A method has been developed for generating search routes for strategic bombers. With certain simplifying assumptions, the routing problem can be modeled as a *covering salesman problem*, for which we have found efficient new heuristics based on spacefilling curves. The result is a two-dimensional route which passes within a specified distance of all the sites of interest, while approximately minimizing the total distance traveled.

This procedure has been extended to include the selection of an appropriate subset of sites when their number is large, so that time and fuel constraints preclude visiting all of them. The finite turn radius is also taken into account.

Problems with both finite sensor range and large numbers of sites have not been treated here, but only for simplicity. Applying both methods simultaneously would require no essential changes.

A bigger effort would be needed to permit revisits of sites, and to properly accumulate credit for multiple visits. Neither the spacefilling curve heuristic nor the strips heuristic easily allows multiple visits. The arbitrary insertion heuristic could accommodate a starting route with multiple visits, and could be adapted to generate such a route as well.

This work still disregards defenses, terrain avoidance, area searches, waypoint offset (so the aircraft does not pass directly over a site), and sensor pointing.

Much of the computational effort of this procedure is in accounting for the finite turn radius of the aircraft. It might be improved by first calculating a lower bound (the length of the piecewise straight line path) and discarding the path if it is far from feasible.

REFERENCES

1. Current, J. R., and D. A. Schilling, "The Covering Salesman Problem," *Transportation Science 23* (3) (1989), 208-213.
2. Garey, M. R., and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, San Francisco: Freeman, 1979.
3. Bartholdi III, J. J., and L. K. Platzman, "An O(*n* log *n*) Planar Traveling Salesman Heuristic Based on Spacefilling Curves," *Operations Research Letters* 1 (4) (1982), 121-125.
4. Platzman, L. K., and J. J. Bartholdi III, "Spacefilling Curves and the Planar Traveling Salesman Problem," PRRC-TR-83-02, Production and Distribution Research Center, Georgia Institute of Technology, 1983.
5. Hilbert, D., "Uber die stetige Abbildung einer Linie auf ein Flachenstuck," *Math. Ann.* 38 (1891).
6. Karp, R. M., and J. M. Steele, "Probabilistic Analysis of Heuristics," in *The Traveling Salesman Problem* (E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, eds.), New York, NY: John Wiley & Sons, 1985, pp. 184, 194.
7. Golden, B. L., "Empirical Analysis of Heuristics," in *The Traveling Salesman Problem* (E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, eds.), New York, NY: John Wiley & Sons, 1985, pp. 217, 223.
8. Bartholdi III, J. J., and L. K. Platzman, "Heuristics Based on Spacefilling Curves for Combinatorial Problems in Euclidean Space," *Management Science* 34 (3) (1988), 291-305.
9. Goldberg, D. E., "Computer-aided Gas Pipeline Operation Using Genetic Algorithm and Rule Learning," (Doctoral dissertation, University of Michigan), *Dissertation Abstracts International 44* (10) (1983).
10. Goldberg, D. E., & M. P. Samtani, "Engineering Optimization via Genetic Algorithm," *Proceedings of the Ninth Conference on Electronic Computation* (1986), 471-482.
11. Etter, D. M., M. J. Hicks, & K. H. Cho, "Recursive Adaptive Filter Design Using an Adaptive Genetic Algorithm," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing 2* (1982), 635-638.
12. Davis, L., & D. Smith, "Adaptive Design for Layout Synthesis," Internal Report, Texas Instruments.
13. Holland, J. H., *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: The University of Michigan Press, 1975.
14. Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Reading, MA: Addison-Wesley, 1989.
15. Goldberg, D. E., & K. Deb, "Genetic Algorithms with Sharing for Multimodal Function Optimization," *Proceedings of the Second International Conference on Genetic Algorithms and their Applications* (1987), Morgan Kaufmann Publishers, Inc., Palo Alto, CA.
16. Beardwood, J., J. H. Halton, and J. M. Hammersley, "The Shortest Path Through Many Points," *Proc. Cambridge Philos. Soc.* 55 (1959), 299-327.
17. Platzman, L. K., and J. J. Bartholdi III, "Spacefilling Curves and the Planar Traveling Salesman Problem," *Journal of the Association for Computing Machinery* 36 (1989), 719-737.
18. Bertsimas, D., and M. Grigni, "Worst-Case Examples for the Spacefilling Curve Heuristic for the Euclidean Traveling Salesman Problem," *Operations Research Letters* 8 (5) (1989), 241-244.
19. DeJong, K., "An Analysis of the Behavior of a Class of Genetic Adaptive Systems," University Microfilms No. 76-9381, University of Michigan, 1975.
20. Grefenstette, J. J., "Optimization of Control Parameters for Genetic Algorithms," *IEEE Transactions on Systems, Man, and Cybernetics* SMC-16 (1) (1986), 122-128.
21. Davis, L., "Adapting Operator Probabilities in Genetic Algorithms," *Proceedings of the Third International Conference on Genetic Algorithms and their Applications* (1989), 61-69, Morgan Kaufmann Publishers, Inc., Palo Alto, CA.
22. Richardson, J. T., et al., "Some Guidelines for Genetic Algorithms with Penalty Functions," *Proceedings of the Third International Conference on Genetic*

Algorithms and their Applications (1989), 191-197, Morgan Kaufmann Publishers, Inc., Palo Alto, CA.

23. Siedlecki, W. & J. Sklansky, "Constrained Genetic Optimization via Dynamic Reward-Penalty Balancing and its Use in Pattern Recognition," *Proceedings of the Third International Conference on Genetic Algorithms and their Applications* (1989), 141-150, Morgan Kaufmann Publishers, Inc., Palo Alto, CA.

A. SPACEFILLING CURVE HEURISTICS FOR THE CSP

Platzman and Bartholdi's generic spacefilling heuristic suggests the following procedure for the Euclidean covering salesman problem: find a heuristic tour through all the cities, then choose a *subset* of cities that will cover all the cities. (Note that solving the routing problem first and the covering problem second is just the opposite of COVTOUR.)

A.1 Basic Algorithms

We have developed a family of heuristics, with different methods of selecting the subset of cities to be visited. The simplest such procedure is as follows:

HEURISTIC CSP-2

a. Use a spacefilling curve heuristic to find a tour through all the points. Denote the sorted points by $p_1 \cdots p_n$, *with corresponding keys* $\theta_1 \cdots \theta_n$.

b. Let $i \leftarrow 1$.

c. Find the last point $p_j \in p_i \cdots p_n$ *for which* $2\sqrt{D(\theta_i,\theta_j)} \le d$. *Mark* p_j (p_j *is the last point that covers* p_i.)

d. Find the first point $p_k \in p_{j+1} \cdots p_n$ *for which* $2\sqrt{D(\theta_j,\theta_k)} > d$. *If there is no such* p_k, *go to f. Otherwise,*

e. Let $i \leftarrow k$ (p_i *is the first point not covered by* p_j) *and go to c.*

f. Visit the marked points in subscript order, then return to the first marked point.

Given the nearness property (8), the tests in steps c. and d. guarantee that every city is covered. However, these tests are conservative by about a factor of two. Somewhat shorter tours can be found by replacing the test $2\sqrt{D(\theta_j,\theta_k)} > d$ in step d. with $|p_j - p_k| > d$, to test the Euclidean distance directly. We will refer to the resulting heuristic as CSP-3. CSP-2 and CSP-3 require $O(n \log n)$ time.

If we made the same replacement in step c., we would know that each city in $p_i \cdots p_j$ could be covered by a visit to p_i, but would not have tested their distances to p_j which is being visited. For $i < i' < j$, even if $|p_i - p_{i'}| \le d$ and $|p_{i'} - p_j| \le d$, it is not necessarily true that $|p_i - p_j| \le d$. However, the last city p_j which covers each city p_i and the intermediate cities, can be efficiently found in a preliminary pass starting with the last point.

HEURISTIC CSP-4

a. Use a spacefilling curve heuristic to find a tour through all the points. Denote the sorted points by $p_1 \cdots p_n$, *with corresponding keys* $\theta_1 \cdots \theta_n$.

b. Let $j \leftarrow n$ *and* $i \leftarrow n$.

c. Let $t_i \leftarrow j$ (p_j *is the last point that covers* p_i), $i \leftarrow i-1$.

d. If $i = 0$ *go to h. Otherwise,*

e. If $|p_i - p_j| \le d$, *go to c. Otherwise,*

f. Let $j \leftarrow j-1$.

g. If $|p - p_j| > d$, *for some point* $p \in p_i \cdots p_j$, *go to f. Otherwise, go to c.*

h. Let $i \leftarrow 1$.

i. Let $j \leftarrow t_i$. *Mark* p_j.

j. Find the first point $p_k \in p_{j+1} \cdots p_n$ *for which* $|p_j - p_k| > d$. *If there is no such* p_k, *go to i. Otherwise,*

k. Let $i \leftarrow k$ (p_i *is the first point not covered by* p_j) *and go to i.*

l. Visit the marked points in subscript order, then return to the first marked point.

If no one city covers more than m other cities, the preliminary pass (steps b. - g.) requires $O(n\,m)$ time. This may dominate the $O(n \log n)$ cost of the sort in step a.

A.2 Extension to Unequal Covering Distances

The formulation of the CSP in [1] permits the covering distance to be different for each city. For example, when routing a rural health care delivery team, one might imagine that the covering distance is a function of the public transportation system at each stop. CSP-4 will still work if d is replaced by d_j. However, CSP-2 and CSP-3 must be modified.

Note, for example, that if $i < j < j'$, we know $2\sqrt{D(\theta_i,\theta_j)} \le 2\sqrt{D(\theta_i,\theta_{j'})}$. Therefore, in step c. of CSP-2, it is sufficient to scan forward to the first city $p_j \in p_{i+1} \cdots p_n$ for which $2\sqrt{D(\theta_i,\theta_j)} > d$, then back up by one. With varying covering distances, this procedure will not necessarily find the last city that covers p_i. As before, we can make a preliminary pass to identify for each city the last city that covers it. The revised procedure is:

HEURISTIC CSP-5

a. Use a spacefilling curve heuristic to find a tour through all the points. Denote the sorted points by $p_1 \cdots p_n$, *with corresponding keys* $\theta_1 \cdots \theta_n$.

b. Let $j \leftarrow n$ and $i \leftarrow n$.

c. If $2\sqrt{D(\theta_i,\theta_j)} > d_j$, go to f.

d. Let $t_i \leftarrow j$ (p_j is the last point that covers p_i).

e. If $i > 1$ let $i \leftarrow i-1$ and go to c. Otherwise, go to g.

f. If $j > 1$ let $j \leftarrow j-1$ and go to c. Otherwise,

g. Let $j \leftarrow t_i$. Mark p_j.

h. Find the first point $p_k \in p_{j+1} \cdots p_n$ for which $|p_j - p_k| > d_j$. If there is no such p_k, go to j. Otherwise,

i. Let $i \leftarrow k$ (p_i is the first point not covered by p_j) and go to g.

j. Visit the marked points in subscript order, then return to the first marked point.

In this case the preliminary pass takes only $O(n)$ time because it is not necessary to test the intermediate cities, so the entire procedure still takes only $O(n \log n)$ time.

A.3 Heuristic Performance

The expected length of the optimum tour through n points uniformly distributed in the unit square is $\beta^*\sqrt{n}$ where $0.765 \le \beta^* \le 0.765 + 4/n$ as shown by Beardwood et al. [16]. The tours generated by TSP-1 are asymptotically bounded above by $\beta\sqrt{n}$,, where $\beta/\beta^* = 1.25$,, and the enhancement in TSP-2 improves this ratio to 1.15 [3].

Platzman and Bartholdi [17] proved that their heuristic yields a tour within a factor $O(\log n)$ of the optimum length. Bertsimas and Grigni [18] have demonstrated that the ratio can in fact be as large as $O(\log n)$ for points arranged along a line. This is also a lower bound on the worst case performance of the heuristics proposed here.

To determine empirically the effectiveness of the different methods, six combined heuristics (TSP-1 and TSP-2 with CSP-2 through CSP-4) were used to solve the same set of 100 CSPs, each with 100 cities distributed randomly in the unit square and a constant covering distance of 0.1. (CSP-5 is not included because it will generate the same tours as CSP-3, if the covering distances are all equal and the keys are unique.) In each case, Sierpinski's curve and Platzman and Bartholdi's $O(n \log n)$ implementation of the improvement procedure were used. The resulting tour lengths are shown in the first two rows of table 2. Entries indicate the mean ± one standard deviation. The benefits of the tour improvement procedure apparently carry over to the resulting CSP tours.

Table 2. Heuristic Tour Lengths for 100 City CSPs

Improve. TSP	Aft. CSP	TSP	CSP-2	CSP-3	CSP-4
no	no	9.62±0.38	9.00±0.41	7.85±0.48	7.17±0.45
yes	no	8.82±0.31	8.24±0.34	7.31±0.37	6.73±0.38
no	yes	9.62±0.38	8.20±0.33	7.10±0.36	6.58±0.36
yes	yes	8.82±0.31	7.97±0.33	6.93±0.34	6.38±0.35

The same improvement procedure can also be applied after selecting the cities to be visited. The results are shown in the last two rows of table 2. As shown in the third row of the table, applying the procedure once, either before or after selecting the cities, results in about the same final tour length. However, applying the procedure before selecting the cities reduces the number of cities in the final tour. The best tours are produced by applying the improvement procedure both before and after selecting the cities. (It can make a further improvement because, with fewer cities present, it will consider larger perturbations.) For the results reported below, the improvement procedure was applied only to the initial TSP tour.

To determine empirically the performance of the heuristics as a function of problem size, TSP-2 combined with CSP-2 through CSP-4 were applied to CSPs of from 20 to 1,000 cities distributed randomly on the unit square, with a constant covering distance of 0.1. The resulting tour lengths are shown in figure 26. Each data point is an average of 50 different CSPs.

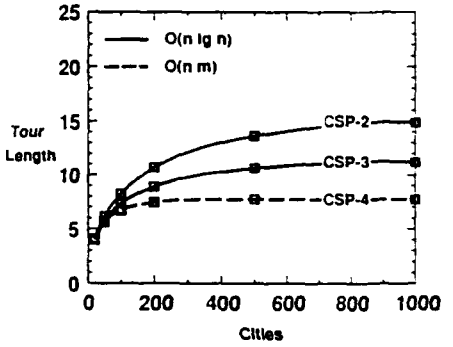


Figure 26. CSP Tour Lengths with Covering Distance $d = 0.1$

Performance statistics for TSP-2 are shown in the first three columns of table 3. For TSPs of 1,000 cities, it generated tours of mean length $27.56 = 0.872\sqrt{1000}$. For a tour of length $0.872\sqrt{n}$, the average distance between successive cities along the tour is $0.872/\sqrt{n}$. For a constant covering distance of d, we would expect the routing part of a CSP to dominate when $d < 0.872/\sqrt{n}$. For larger n, the set covering part of the problem would dominate. For $d = 0.1$ the boundary between the two regions would be at $n_b = (0.872/0.1)^2 = 76$. This effect is clearly evident in figure 26.

TSP-2 combined with CSP-2 through CSP-5 were then applied to the same set of CSPs, but with covering distances chosen uniformly from the interval [0,0.1]. The resulting tour lengths are shown in figure 27. For this case, the mean covering distance is $\bar{d} = 0.05$,, so the boundary between the routing- and set covering-dominated regions is at approximately $n_b = (0.872/\bar{d})^2 = 304$.

Table 3. Performance of Different Heuristics

TSP			CSP				
CITIES VISITED	TOUR LENGTH	CPU TIME	d	METHOD	CITIES VISITED	TOUR LENGTH	CPU TIME
20	4.13±0.39	0.09	0.1	CSP-3	16.14±1.55	3.99±0.40	0.002
50	6.32±0.35	0.24	0.1	CSP-3	31.52±2.34	5.80±0.38	0.006
100	8.81±0.30	0.50	0.1	CSP-3	47.04±3.19	7.35±0.38	0.007
200	12.37±0.32	1.03	0.1	CSP-3	64.12±3.36	8.95±0.45	0.020
500	19.55±0.36	2.66	0.1	CSP-3	85.48±3.63	10.64±0.44	0.044
1000	27.56±0.27	5.44	0.1	CSP-3	95.88±3.68	11.19±0.44	0.093
20	4.13±0.39	0.09	0.1	CSP-4	15.84±1.60	3.99±0.40	0.004
50	6.32±0.35	0.24	0.1	CSP-4	28.82±2.23	5.59±0.36	0.012
100	8.81±0.30	0.50	0.1	CSP-4	39.62±2.78	6.72±0.37	0.018
200	12.37±0.32	1.03	0.1	CSP-4	48.10±3.27	7.46±0.47	0.043
500	19.55±0.36	2.66	0.1	CSP-4	57.70±2.30	7.74±0.34	0.118
1000	27.56±0.27	5.45	0.1	CSP-4	62.86±1.71	7.72±0.27	0.275
20	4.13±0.39	0.09	[0,0.1]	CSP-5	18.26±1.24	4.10±0.38	0.002
50	6.32±0.35	0.25	[0,0.1]	CSP-5	40.24±2.63	6.12±0.33	0.006
100	8.81±0.30	0.51	[0,0.1]	CSP-5	68.96±4.11	8.13±0.33	0.012
200	12.37±0.32	1.05	[0,0.1]	CSP-5	105.22±5.73	10.44±0.39	0.024
500	19.55±0.36	2.70	[0,0.1]	CSP-5	159.56±9.38	13.63±0.58	0.059
1000	27.56±0.27	5.54	[0,0.1]	CSP-5	178.68±9.46	15.07±0.55	0.117
20	4.13±0.39	0.09	[0,0.1]	CSP-4	17.80±1.28	4.08±0.39	0.003
50	6.32±0.35	0.25	[0,0.1]	CSP-4	37.26±2.91	6.03±0.36	0.007
100	8.81±0.30	0.51	[0,0.1]	CSP-4	59.10±3.62	7.75±0.34	0.014
200	12.37±0.32	1.05	[0,0.1]	CSP-4	81.64±5.90	9.54±0.51	0.030
500	19.55±0.36	2.70	[0,0.1]	CSP-4	101.76±5.62	11.06±0.45	0.083
1000	27.56±0.27	5.54	[0,0.1]	CSP-4	103.38±6.29	10.98±0.47	0.185

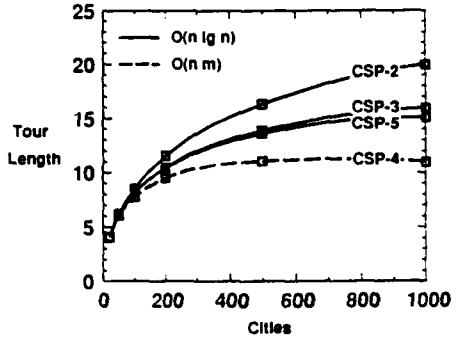


Figure 27. CSP Tour Lengths with Covering Distance d_j Chosen Uniformly from [0,0.1]

Performance statistics for some of these runs are shown in table 3. For tour lengths and number of cities visited in the CSP tour, the mean and one standard deviation are listed. The execution times (in seconds for an 80386-based personal computer) for the routing and the set covering parts of the CSP are listed separately. CSP-4 took somewhat more time than CSP-5, but the time for TSP-2 dominated in each case, and the actual times were approximately proportional to n.

A.4 Remarks

The effectiveness of these heuristics is due to the fact that the spacefilling curve is *self similar*, so that a subsequence of a heuristic tour is itself a heuristic tour [17]. This is not true of some heuristic solutions to the TSP, such as the "strips" method described by Karp and Steele [6] or the nearest-neighbor heuristic.

These heuristics for the CSP have disregarded the fact that both the tour and the spacefilling curve ϕ are closed. Suppose a tour includes p_{j_2} which covers p_j and p_{j_3} and p_{j_4}, which covers no other cities. By the nearness property (8) we expect p_{j_4} to be close to p_j. If p_{j_2} also happened to cover p_{j_4}, the tour could be shortened by omitting p_{j_4}.

For $n >> n_b$, very little benefit was found in using TSP-2 rather than TSP-1 . Apparently, most of the local changes made by the improvement procedure are lost when covered cities are removed from the tour. An alternate interpretation is that the routing part of the problem is comparatively unimportant in those cases, and the primary benefit of a spacefilling curve heuristic is to linearize the underlying SCP. This leads us to the recognition that the same heuristics can be applied to planar SCPs with Euclidean metrics. The result would then be the set of marked points, and their order would be disregarded. This heuristic would be approximate in more than one sense. Even the optimal (minimum tour length) solution to the CSP would not necessarily correspond to an optimal (fewest cities) solution to the SCP.

B. ADAPTIVE OPERATOR PROBABILITIES

The efficiency of a genetic algorithm depends on the settings of several parameters including population size and the probabilities of applying each genetic operator. DeJong found robust settings by using a test suite of function optimization problems [19]. Later, Greffenstette used a genetic algorithm to find a better set of parameter values [20]. However, these studies used only the crossover and mutation operators.

When additional operators are used, it is necessary to set their selection probabilities. Setting them by either of the above methods would require a very considerable effort. It is attractive to set them adaptively instead.

The following scheme for adaptively setting operator probabilities was presented by Davis [21]. Each new individual that is better than the current best individual acquires a "local delta" equal to the improvement. At intervals of I evaluations, each of the last W individuals pass back to its parent(s) P times the sum of its local delta and the deltas inherited from its progeny (except that no delta is passed back more than M generations). Each individual then passes its remaining delta to the operator which created it. The operators accumulate the deltas in a vector C. The vector V of operator probabilities is updated by shifting a fraction S, so that $V \leftarrow SC/|C|+(1-S)V$.

The five parameters W, I, S, P, and M effectively replace the operator probabilities as governing parameters. Davis found the values $W = 100$, $I = 50$, $S = 0.15$, $P = 0.15$, and $M = 10$ to be effective.

There are two points that seem to make implementation of this awkward. The first is that even deleted members would have to be kept around, on the chance that one of their progeny would pass back some delta. The other concerns the handling of M, the limit on the number of generations that delta is passed back. If M could be disregarded, the deltas could be propagated in one pass, starting with the most recent member, with each step involving only the member's immediate parents. Accounting for M would seem to require a walk of the family tree of each member having a local delta, to a depth of M or the edge of the adaptation window (whichever came first). This procedure would seem to grow exponentially with W/population, and would be particularly lengthy if two-parent operators were successful (leading to large family trees).

In addition, one might want to reward operators when they created individuals that were better than any of their parents, regardless of whether they were better than the previous best individual. This would give the adaptation mechanism more information to work with.

Because of the above concerns, the following somewhat simpler scheme has been used instead.

Each member has an associated "responsibility vector" R which indicates the relative roll each genetic operator had in its creation. If operator g_k is used to create child c from a single parent p, the new member gets a responsibility vector

$$R_c = PR_p + (1-P)U_k \qquad \text{(B-1)}$$

where U_k has a 1 in the kth position and 0s elsewhere. If the operator used two parents m and f, the child gets a responsibility vector

$$R_c = P(R_m + R_f)/2 + (1 - P)U_k \qquad \text{(B-2)}$$

The operators accumulate credit for improvements in a vector C. If the fitness of the new member exceeds that of its better parent by δ, C is incremented by $\delta R_c/|R_c|$. If the new member is less fit than at least one parent, C is not changed. Each generation (m births), the vector V of operator probabilities is updated by shifting a fraction S:

$$V \leftarrow SC/|C|+(1-S)V \qquad \text{(B-3)}$$

and the credit vector is attenuated:

$$C \leftarrow \mu C \qquad \text{(B-4)}$$

For this work, $P = 0.90$, $S = 0.30m/100$, and $\mu = 0.50$.

This scheme follows the two principles suggested by Davis, namely that the probability of applying an operator should be altered in proportion to the performance of the individuals it creates, and that local measures of performance are insufficient (that is, operators creating ancestors of successful individuals should also be rewarded).

An example of operator probability adaptation is shown in figure 28. Each curve is an average over four runs. The adaptive procedure confirms the importance of the crossover operator, since its probability steadily increases. On the other hand, it suggests the "move-bit" operator may be relatively unhelpful (or perhaps helpful only early in the optimization process).

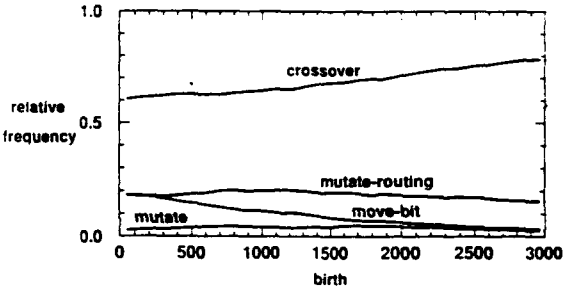


Figure 28. Operator Probability Adaptation

C. CONSTRAINED OPTIMIZATION

Suppose we wish to use a genetic algorithm to minimize the function

$$f(x) \qquad \text{(C-1)}$$

while satisfying the constraint

$$g(x) < 0\,. \qquad \text{(C-2)}$$

Goldberg suggests a penalty function approach so that the objective function is replaced by a compound objective function:

$$f(x) = f(x) + \alpha\Phi(z) \qquad \text{(C-3)}$$

where $z = g(x)$, α is a nonnegative *penalty coefficient* and Φ is a nonnegative *penalty function* [14].

Richardson, *et al.*, found that a penalty function should not be too harsh because the most direct path from a given solution to the optimal solution may require some infeasible intermediate structures [22]. That is, the penalty function should be chosen to balance the preservation of information against the pressure for feasibility.

It is necessary to penalize infeasible members, but not so much that moderately infeasible members cannot contribute information. It is helpful to reward moderately feasible members for satisfying the constraint well. The penalized fitness function should have a maximum at the *feasibility boundary* (where $g(x) = 0$). Thus, we want a penalty function which is smooth at 0 and has a slope there equal to f'/g'. On the other hand, members well inside the feasibility boundary should be neither penalized nor rewarded, so the penalty function should be zero for large negative values. Members well outside the feasibility boundary should be heavily penalized, so the penalty function should be large for large positive values.

One effective penalty function is

$$\Phi(z,w) = ze^{z/w} \qquad \text{(C-4)}$$

which is shown with a solid line in figure 29.

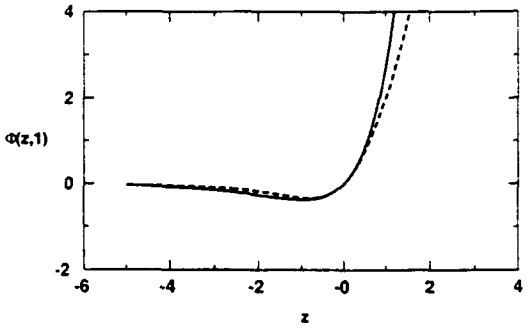


Figure 29. Penalty Functions

A simpler alternative which works almost as well is

$$\Phi(z,w) = \begin{cases} \dfrac{z}{\left(1-\dfrac{z}{w}-\dfrac{z^3}{w^3}\right)}, & \text{for } z \le 0 \\ z\left(\dfrac{z}{w}+1\right), & \text{for } z > 0 \end{cases} \qquad \text{(C-5)}$$

which is shown with a dotted line in figure 29. Each function has a unit slope at z=0, so the compound objective function will have a local minimum at the feasibility boundary if $\alpha = f'/g'$. α then plays the role of a Lagrange multiplier. Each function also has a second parameter w, a characteristic width, which is discussed below.

Siedlecki and Sklansky introduced the concept of *dominance* [23]. Recall that we have assumed a minimization problem, so that better solutions correspond to smaller values of f. Solution x_i *dominates* solution x_j if $f_\alpha(x_i) < f_\alpha(x_j)$ for any positive value of α. Evidently, this implies both

$$f(x_i) < f(x_j) \qquad \text{(C-6)}$$

and

$$g(x_i) < g(x_j). \qquad \text{(C-7)}$$

If α is too large the optimal solution may be precluded, and if α is too small then members will tend to be infeasible. It is therefore important that α be close to f'/g'. This ratio is estimated from the current population (specifically, from dominant members on either side of the feasibility boundary) using a single state Kalman filter as follows:

DERIVATIVE ESTIMATE

a. *Sort the members on their constraint values.*
b. *Find a fit member near the feasibility border (the most fit member x_f with constraint $-1 < g(x_f) < 0$)*
c. *Find an infeasible member near the feasibility border (the most fit member x_i with constraint $0 < g(x_i) < 1$).*
d. *If the moderately infeasible member was more fit (that is, $f(x_i) < f(x_f)$), the derivative f'/g' is estimated as $\alpha' = (f(x_i)-f(x_f))/(g(x_i)-g(x_f))$. The filtered derivative estimate is updated according to $\alpha \leftarrow (P\alpha'+R\alpha)/(P+R)$, and its estimated covariance P is updated according to $P \leftarrow RP/(R+P)+Q$*
e. *Otherwise, $P \leftarrow P+Q$*

Q, R, and P are initialized in the ratio $1:10:10^{11}$, so the *Kalman filter gain* $P/(P+R)$ is very high on the first iteration, drops quickly thereafter, and doesn't rise again unless there are several generations in a row with no improvements.

For some runs, the width w was kept constant at 2. However, there was some evidence that decreasing w made the accurate determination of α less critical. This was done by initially setting w to 1, and thereafter to the range of the constraint function for the current population:

$$w \leftarrow z_{max} - z_{min} \qquad \text{(C-8)}$$

AN EFFICIENT METHOD FOR THREE-DIMENSIONAL ROUTE PLANNING WITH DIFFERENT STRATEGIES AND CONSTRAINTS

U. Leuthäusser
F. Raupp
ESG Elektronik System Gesellschaft mbH
Vogelweideplatz 9
D-8000 München 80
Germany

Summary

A method for planning paths through a digital map is presented. The task consists of planning 3-dimensional paths for autonomous air vehicles over a terrain represented by a digital map with different strategies and constraints. Possible strategies minimize time, danger, or fuel consumption. Constraints are lateral and vertical maximum accelerations as well as time and fuel constraints.

The path planning problem can be viewed as a shortest path problem (geodesic problem) in a space with a non-Euclidean metric depending on the applied strategy. One has to minimize a cost functional giving rise to the Hamilton-Jakobi equation which is solved by Dynamic Programming techniques. Because of the digitization bias, a second optimization step consisting of direct optimization methods is sometimes necessary. The result is a fast, non-heuristic and flexible technique for strategic route planning.

1 Introduction

One fundamental problem for an autonomous vehicle or aircraft is to plan a path from an initial state to the final state of the vehicle in or over a known terrain. The state of the robot consists in general of positon, orientation, time, speed etc. A survey of recent developments in this field is given by Mitchell (Ref 1) with helpful references.

In this paper terrain (represented by a digital map) is either a superposition of natural environment with artificial obstacles and forbidden regions or a 'danger map' derived from a digital terrain map.

The planning task takes place under certain strategies, for example minimization of mission duration or of fuel consumption, minimization of threat, or combinations of such strategies. Additionally, constraints like lateral and vertical maximum accelerations and of course collision freeness are to be taken into account.

The goal of this paper is to give a unified approach to the path planning problem mapping it to a geodesic problem in a curved geometric space. Links to other approaches are pointed out. Furthermore we present an efficient method consisting of several succesive steps for the numerical solution of the path planning problem.

2 Equations for path planning and calculation methods

One approach in planning paths for an autonomous vehicle through a map is to minimize a cost function describing the "costs" along a path. Costs might be energy, safety, time etc. This variational problem is equivalent to finding the shortest paths (geodesics) in a non-Euclidean space with a metric depending on the applied strategy.

The existence of a metric means that the length of a path $x(t)$ can be expressed by the integral

$$J = \int_a^b ds = \int_a^b \left(\sum_{i,k=1}^{n} g_{ik}(x_j, \dot{x}_j)\, \dot{x}_i\, \dot{x}_k \right)^{1/2} dt \qquad (1)$$

where the g_{ik} are the components of the (symmetric) metric tensor.

Setting the first variation of J equal to zero, one obtains the Euler-Lagrange equations. These second order differential equations are not convenient to treat boundary value problems with given endpoints instead of initial conditions.

A different approach uses an equation for S, defined as the optimal J, i.e. as the integral of (1) evaluated along the optimal trajectory. S is called action in mechanics, eikonal in optics. Here it has the meaning of the geodesic distance. For S a non-linear partial differential equation can be derived, the so-called Hamilton-Jacobi (HJ) equation (Ref 2,3).

In the case of x-dependent g_{ik}, the expression under the integral sign of (1) is a homogeneous function of degree one in $\dot{x}_i$. J does therefore not change its form under a transformation of the parameter t. Using this relation one obtains (Ref 2)

$$\sum A_{ik} S_{x_i} S_{x_k} = 1 \qquad (2)$$

where the matrix A_{ik} is the inverse of matrix g_{ik} of (1). Equation (2) has to be solved with the boundary condition $S(x_{goal}) = 0$. If S is known, one can construct the optimal paths from all points in state space to x_{goal}.

There are several links to the path planning problem mainly with an origin in physics which reveal the fundamental structure of the problem. Fermat's principle as the optical analogue can be expressed by (1) with a diagonal metric tensor g_{ii} proportional to the squared defraction index. A mechanical analogue is obtained using the position dependent kinetic energy as g_{ii}.

In (Ref 10) a diffusion model is applied to path planning (including potential field methods of Ref 11 and 12). It is shown in Ref. 4 that such an approach can also be reduced to a geodesic problem. The diffusion equation in an external potential which can be transformed to a Schrödinger equation is in the small diffusion limit equivalent to a Hamilton-Jakobi equation (Fig. 1).

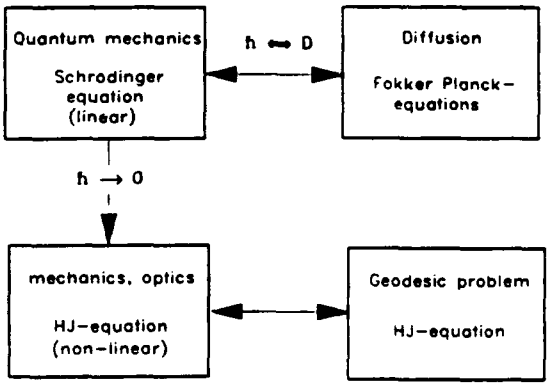


Figure 1: Relations between fundamental equations.

Apart from some analytical approaches (Ref 4) the most efficient method for the treatment of the HJ-equation is Dynamic Programming. Bellman's Dynamic Programming technique (Ref 5,6) converts the optimization problem into a problem of solving a set of recursive equations and is the first step of our solution method.

In a discrete-time formulation one has

$$S(\mathbf{x}) = \min_{u} \{ J(\mathbf{x}, \dot{\mathbf{x}} = u) \} \qquad (3)$$

where

$$J(\mathbf{x},u) = \sum_{k=0}^{K} L(\mathbf{x}(k),u(k))$$

is the analogue of (1) with L as path element ds. Note that in our application L has no explicit k-dependence.

The n-dimensional vector u(k) determines the propagation of $\mathbf{x}(k)$

$$\mathbf{x}(k+1) = \mathbf{x}(k) + u(k) \qquad (4)$$

into the various directions of planning space.

Because of the approximation to choose only a finite number of u's, the continuous planning space is approximated by a lattice (typically a 3-d square lattice) als called grid graph. Start and goal are now any two nodes in the lattice.

The functional equation of Dynamic Programming can now be derived and is given by

$$S(\mathbf{x}(k),k) = \min_{u} \left\{ L\left[\mathbf{x}(k),u(k),k\right] + S\left[\mathbf{x}(k)+u(k),k+1\right] \right\} \qquad (5)$$

Expanding the right hand side of (5) in a Taylor series about $\mathbf{x}(k)$ and executing the minimum operation one obtains expressions for u, which determine the HJ-equation (2). It turns out, however, that the recursive form of equation (5) is easier to cope with.

As usual in Dynamic Programming , the recurrence relation (5) has to be solved backward in k. The minimum procedure gives a direction matrix $u(\mathbf{x},k)$ which gets k-independent in about $(\mathbf{x}^2_{max})^{1/2}$ iterations (the k-independence of u is the criterium to stop the algorithm), where the $\mathbf{x}_{max}$ are the maximum lengths in the various directions of the planning space.

The direction matrix $u^*(\mathbf{x})$ determines all optimal paths of the complete search space to one goal point $\mathbf{x}_{goal}$, which is characterized by $S(\mathbf{x}_{goal},k) = 0$ for all k, whereas all other S have to be set to infinity at the beginning of backward iteration.

Apart from the discretization errors, the final $S(\mathbf{x})$ would be sufficient to solve the variational problem. For the special case treated here, the optimal paths are perpendicular to the contours of constant S, so that the gradient of S determines the path.

3 Strategies and cost functions

For our planning task to find the optimal path of a missile from the launch-point to a fixed goal we introduced strategies like safety, speed, and fuel consumption or mixtures of this strategies.

Flying as low as possible is considered to be a measure for the saftey of a missile to avoid unknown threats. For a strategy maximizing the safety and minimizing the duration of flight an appropriate cost function was assumed. This function is given by the metric tensor

$$g_{ij} = \delta_{ij} \cdot \left(1 + \frac{z(t)}{R}\right)^2 \qquad (6)$$

The vertical component z(t) forces the aircraft to fly always near the surface. The parameter R determines the ratio between speed and safety.

The terrain V(x,y) the missile has to pass is a superposition of the natural environment and artificial obstacles. The additional contributions to the natural terrain may be forbidden regions of military threats, areas with higher

degrees of danger etc. V(x,y) is then represented by a digital map of an effective terrain.

To solve the variational problem we have to take into account some boundary conditions. Using the effective terrain we get the boundary condition

$$z(t) \geq V(x(t),y(t)) \qquad (7)$$

To fulfill this condition a penalty function can be added to the g_{ij} of (6):

$$\delta_{ij} \cdot \alpha \cdot \theta(V(x,y) - z(t)) \qquad (8)$$

The stepfunction $\theta = 1$ if z(t) is below V(x,y) and 0 otherwise. α is a parameter weighting the strength of the penalty.

Other boundary conditions have to be introduced because of the limitations imposed by the flight dynamics of an aircraft. To get allowed flight paths the following constraints have to be taken into account:

$$|\ddot{z}(t)| \leq a_z \qquad (9)$$

a_z is the maximum possible acceleration in the z-direction. The minimum horizontal curvature of the flight path can be expressed as follows

$$\left((\ddot{x}(t))^2 + (\ddot{y}(t))^2 \right)^{1/2} \leq a_{xy} \qquad (10)$$

with a_{xy} representing the lateral acceleration.

The boundary conditions (9) and (10) can be treated with penalty functions similiar to (8). Using a discrete search space in the Dynamic Programming procedure allows another possibility to implement the constraints imposed by the flight dynamics. These constraints limit the number of directions u(k) at each node of the discrete space. The allowed directions are determined by the flight dynamics.

These boundary conditions also restrict the size of the lattice. To exclude impossible movements of the aircraft a minimum distance between adjacent nodes is required. The optimal resolution of the digital map is therefore determined by this minimum distance.

If the terrain allows the missile in any case to follow the surface V(x,y) we can reduce exactly the task of planning on a 3-D lattice to a 2-dimensional search. Therefore in the case of a terrain which is not too ragged we separate the variational problem in two independent steps. The first step is the search for the optimal path on the surface V(x,y). In the second step z(t) is calculated along the path found in the prior step.

This method of planning in two independent steps has the great advantage of reducing the calculation effort. However, its application is limited. Planning in terrains showing steep ascents the 2-step solution might fail because the aircraft cannot completely follow (because of the acceleration constraints) the terrain. In this case the optimal paths have to be calculated on a 3-dimensional lattice.

To minimize fuel consumption one can use in a first approximation

$$g_{ik} = \delta_{ik} \cdot (\alpha x^2 + \beta \cdot (\nabla V) \cdot x \cdot \theta[(\nabla V) \cdot x]) \qquad (11)$$

where θ is the step function.

The space of possible paths between two endpoints has a complicated structure, in general with many local minima. If the search with Dynamic Programming is not too crude, one finds with this method the region of attraction of the global minimum and has therefore a convex minimum problem, which can be treated in a subsequent step using the primary solution delivered by Dynamic Programming as the initial value.

In the discrete formulation for J one has a normal minimum problem

$$J(x,u) = \sum_{k=0}^{K} L(x(k),x(k+1)-x(k)) = \min \qquad (12)$$

for the x(k) which can be treated by first or second order gradient methods. It is useful to perform a Fourier transformation of the x(k). Because of the smoothness of the flight paths the Fourier series can be limited to 15 terms maximum, reducing considerably the number of variables which are now the Fourier coefficients. The expected gain in J is of the order of a few percent.

4 Extensions and reduction of calculation effort

An important point is the reduction of the calculation effort by the constraints in the space X(k) of the admissible x(k)

$$X(k) = \left\{ x(k) \;\; | x(k) \equiv x(k+1) + u(k+1) \right.$$

$$\left. \text{with} \quad x(k+1) \in X(k+1) \quad \text{and} \quad x(k+1) \neq x(k+2) \right\}$$

X(k) contains only those points x(k) which are neighbours of points which have changed in the previous step, or equivalently, only those points or nodes x(k+1) have to be expanded which have been changed. Without this extension, the basic Dynamic Programming algorithm would be about a factor 20 slower.

An additional constraint comes from policy space U(x,k). U(x,k) contains only admissible directions of motion (for example induced by the allowed maximal accelerations of a vehicle) and depends therefore from policy space of the previous stage U(x,k+1).

Compared to A* and Dijkstra's algorithm (Ref 7,8), the described algorithm avoids the search of the minimum element in set "OPEN", with the advantage to get all optimal paths to a certain goal in a wave propagation type of search. Note that in contrast to Dijkstra's and A*-algorithm, a node can be calculated several times in order to get the optimal path.

Further increase of efficiency can be obtained by Branch-and-Bound methods. With this method (Ref 9) the set of lattice points which have to be considered can be reduced, still getting exact results. The idea proceeds as follows:

A node i for which

```
cost_at_time_n (goal,node i)  +
cost_lower_bound (node i,start)  >
cost_upper_bound (goal,start)
```

holds, needs not to be expanded at time step n because an optimal path including that node i is impossible (this follows immediatley from Bellman's principle of optimality). An upper bound on the cost function between start and goal can be obtained by perturbation methods. The total number of node calculations can be reduced by about 30 percent.

The complexity of the described algorithm is linear in the nodes. Suppose a d-dimensional lattice with a total number of nodes n. The number of time steps of the algorithm until the final direction matrix is found is of order $O(n^{1/d})$. The number of nodes which have to be calculated in one time step lie in a (d-1)-dimensional wavefrontlike hypersurface with nodes of order $O(n^{1-1/d})$ which yields a total complexitiy O(n).

5 Applications

The described method of route planning can be exploited for many problems of autonomous robot movement. Examples are finding routes in artificial environments or calculating collision free paths of robot arms. The applicability of the technique to problems of this type and its effectiveness has been demonstrated (Ref 4).

The algorithm was mainly developed for planning optimal paths of autonomous missiles using digital maps. The following features are realized in our planning system:

- *Different strategies*
  - safety
  - minimal duration of flight
  - minimal fuel consumption
  - mix of the above strategies.
- *Danger avoidance*
  By superposition of regions of danger and the natural environment artificial danger maps are created, including:
  - absolutely forbidden areas of different shape
  - areas with z-dependent degrees of danger
  - shaded regions within the range of radar waves.
- *Aircraft flight dynamics*
  The characteristics of an aircraft like drag and lift coefficient, reference wing area and thrust are used to calculate performance values of the missile flying along a selected path:
  - velocity of the missile during different states of flight (horizontal, curve, climbing)
  - fuel consumption
  - constraints coming from flight dynamics:
    maximum angle of climb,
    minimum radius of curvature.
- *Mandatory navigation points*
  option for example to select points the missile has to pass for navigation updating.

6 Results

The result is an efficient technique for strategic route planning using a combination of Dynamic Programming and direct optimization methods. the technique allows the processing of complex cost functions of different nature. It is applied to military flight planning using digital maps.

The algorithm is implemented on a PC-AT with a 386 processor. We have tested our method by planning flight paths in different terrains using digital maps covering regions of up to 200 x 200 km. For the planning task we disretized the maps creating grids of 50 x 50 nodes. In the case of 3-dimensional planning we used 50 x 50 x 12 lattice points.

We obtained the following times to calculate an optimal flight path:

2-step method $\rightarrow$ 12 - 15 sec
3-D planning $\rightarrow$ 80 - 90 sec.

In future, the performance will be further increased by parallelizing the described algorithm and by the use of specialized hardware.

References

1. Mitchell, J.S.B., "An Algorithmic Approach to some Problems in Terrain Navigation", Artificial Intelligence 37, 1988, 171.
2. Courant, R., Hilbert, D., "Methoden der mathematischen Physik I, II", Springer-Verlag, Berlin-Heidelberg-New York, 1968.
3. Bryson, A.E., Ho, Y.C., "Applied Optimal Control", Hemisphere Publishing Co., New York, 1975.
4. Leuthäusser, U., "A Method for Path Planning Based on a Combination of Variational Calculus and Direct Optimization Methods", SPIE Proceedings, Vol.1613, 1991, to be published.
5. Bellman, S.E., "Dynamic Programming", Princeton University Press, 1957.
6. Larson, R.E., Casti, J.L., "Principles of Dynamic Programming, Part II", Marcel Dekker Inc., New York.
7. Nilsson, N.J., "Principles of Artificial Intelligence", Tioga Publishing Co., 1979.
8. Papadimitriou, C.H., Stieglitz, K., "Combinatorial Optimization", Prentice-Hall Inc., 1982.

9. Morin, T.L., Marsten, R.E., "Branch-and-Bound Strategies for Dynamic Programming", Operations Research 24, 1975, 611.
10. Schmidt, G., Neubauer, W., "High-Speed Robot Path Planning in Time-Varying Environments Employing a Diffusion Equation Strategy", in The European Robotics and Intelligent Systems Conference EURISCON, 1991.
11. Khatib, O., "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", Int. Journal of Robotics Research 5.1, 1986, 90.
12. Connolly, C.I., Burns, J.B., Weiss, R., "Path Planning Using Laplace's Equation", Proc. IEEE Int. Conf. on Robotics and Automation, 1990, 2102.

OPTIMAL GUIDANCE ANTICIPATING MISSILE PERFORMANCE

W. Grimm
K.H. Well
Institute for Flight Systems Dynamics
German Aerospace Research Establishment DLR
8031 Weßling, Federal Republic of Germany

SUMMARY

Two different guidance methods are designed for the pre–launch phase in a one–versus–one air combat situation with missiles. In both guidance schemes a missile/target simulation is performed to estimate the miss distance at the end of a fictitious missile launch. A way to do this in real time is described. The first guidance method continuously evaluates the miss distance for each aircraft against the respective opponent. The guidance is such that the ratio of the rates of the miss distances takes an optimal value from the view of the guided aircraft. Thus, it reaches a firing opportunity first.

The purpose of the second guidance method is to reach a firing position against a nonmaneuvering target in minimum time. Inside the guidance algorithm the optimal pre–launch trajectory is approximated by nonlinear programming. It serves as reference trajectory for the guidance. In simulations the influence of different missile and target strategies in the post–launch phase is examined. The comparison with an optimal trajectory shows the near–optimal character of the guidance method.

LIST OF SYMBOLS

D	aerodynamic drag
g	gravitational acceleration (constant), set to 9.80665 m/s^2
h	altitude
n	load factor
q	dynamic pressure
R	miss distance at the end of a missile/target simulation
t	time
T	thrust
V	velocity
W	weight
x	downrange
y	crossrange
z	state vector, $z = (x, y, h, V, \gamma, \chi)$
γ	flight path angle
μ	bank angle
ξ	power setting
χ	heading angle

subscripts:

min	minimum value
max	maximum value
0	initial value
f	final value
i	aircraft No. i, i = 1,2
T	target variable
M	missile variable

superscript:

r	reference value

1. INTRODUCTION

The topic of this paper are optimal guidance schemes in the pre–launch phase. We consider a combat situation between two fighter aircraft, say A1 and A2, each equipped with an advanced medium–range air–to–air missile. Optimal guidance of A1 means that A1's control is optimal in the sense of some performance index. The optimal control formulation involves missile performance at the current position or at some point on a reference trajectory. This means to "anticipate missile performance".

To evaluate missile performance there must be a criterion to decide if a certain state provides a firing opportunity for A1 and/or A2. A first idea is to use "firing envelopes", i.e. surfaces in the state space, which envelop the set of successful firing positions of A1 and A2, respectively. Menon and Duke (Ref. 1), for instance, invented a simple model for the firing envelope. However, a closed–form representation of the firing envelope will never be a realistic model. Note that the dimension of the joint state space is at least 10 if point mass models for A1 and A2 are assumed.

In this paper missile/target simulations are performed on–line to identify a firing position. This was already proposed by other papers related to the present topic (Refs. 3–6, 11). A real–time simulation must be adapted to real–time conditions. The present paper suggests a simulation model which needs much less computational effort than conventional trajectory simulation.

The outcome of a missile/target encounter depends on both the missile and the target guidance. In the simulation it is assumed that the missile is guided by Proportional Navigation and the target performs near–optimal missile avoidance. The guidance law for missile avoidance was derived by Shinar and Gazit (Ref. 2). The output of the simulation is the distance R between missile and target measured at a specified closing speed. The missile is assumed to hit the target if R is below a certain bound.

In reality the target will not detect the missile before its autonomous phase. Only then the target will switch to missile avoidance. This effect is not modeled in the present study; we rather assume "fire–and–forget" missiles.

Having a module to evaluate missile performance in real time, optimal guidance schemes for A1 are designed under different assumptions about A2:

a) A2 behaves aggressively. In this case it is crucial to reach a firing opportunity first.
b) A2 flies straight on without maneuvering. A firing position against A2 is to be reached in minimum time.

A feedback guidance for a) is outlined in the present paper. First results are described by Prasad, Grimm, Berger (Ref. 6).

A guidance method for b) internally computes the optimal pre–launch trajectory. The technique of Shinar and Spitzer (Ref. 7) is to precompute a set of optimal flight paths and to select the element fitting to the current state. The present approach is an on–line approximation of the optimal trajectory via nonlinear programming. The method is described by Grimm (Ref. 8) for the case that the terminal constraint is a condition on the Euklidean distance. In the present paper the Euklidean distance is replaced by the miss distance R, which results from a missile/target simulation starting at the final position of interceptor and target. As in Ref. 8 the guidance method proves to be near–optimal in terms of flight time. A simulated trajectory of A1 lies in the neighbourhood of the optimal one for the same boundary conditions.

2. VEHICLE MODEL

Each vehicle (aircraft or missile) is considered as a point mass accelerated by thrust, weight and aerodynamic forces. Assuming
- a flat, nonrotating earth,
- absence of side force,
- absence of wind and
- a negligible influence of the angle of attack on to the direction of the forces,

the following equations of motion hold:

$$\dot{x} = V \cos\gamma \cos\chi \tag{1}$$

$$\dot{y} = V \cos\gamma \sin\chi \tag{2}$$

$$\dot{h} = V \sin\gamma \tag{3}$$

$$\dot{V} = g\left[\frac{T - D(h,V,n)}{W} - \sin\gamma\right] \tag{4}$$

$$\dot{\gamma} = g\,(n\cos\mu - \cos\gamma)/V \tag{5}$$

$$\dot{\chi} = g\,n\sin\mu\,/\,(V\cos\gamma) \tag{6}$$

The load factor n and the bank angle μ are the controls of the system. n is the ratio of lift and weight; lift is the product of dynamic pressure q, wing reference area S and lift coefficient C_L:

$$n = \frac{q\ S\ C_L}{W} \tag{7}$$

q is defined as

$$q = \rho(h)\ V^2 / 2, \tag{8}$$

where ρ denotes air density. Drag is structured as

$$D(h,V,n) = q\ S\ C_D(M, C_L)\ . \tag{9}$$

The drag coefficient C_D depends on the Mach number M and the lift coefficient C_L. According to (7) C_L is replaced in (9) by

$$C_L = \frac{n\ W}{q\ S}\ .$$

Thus, D actually becomes a function of h, V and n. The load factor is bounded by the

"structural limit" and the maximum lift coefficient $C_{L,max}(M)$:

$$n \le n_{max}, \tag{10}$$

$$n \le (q\ S\ C_{L,max}(M))/W. \tag{11}$$

The aircraft is subject to the terrain and dynamic pressure constraint:

$$h \ge 0 \tag{12}$$

$$q \le q_{max} \tag{13}$$

Aircraft weight is assumed to be constant. Mass reduction due to fuel flow is neglected since we only consider short maneuvers. Aircraft thrust is a function of altitude, speed and the "power setting" ξ:

$$T(h,V,\xi) = T_{min}(h,V) + \xi \cdot (T_{max}(h,V) - T_{min}(h,V)) \tag{14}$$

T can continuously be adjusted between its minimum and maximum value. Accordingly, ξ is bounded between 0 and 1:

$$0 \le \xi \le 1 \tag{15}$$

The aircraft model functions

$$T_{min},\ T_{max},\ C_D,\ C_{L,max}$$

are analytic functions approximating table data of a F4 aircraft model.

Missile thrust is a piecewise constant function of time:

$T = 22760$ N,	$0 \le t \le 3$s ("boost phase").
$T = 5400$ N,	$3\text{s} < t \le 15$ s ("march phase").
$T = 0$,	$t > 15$ s ("coasting phase").

Missile weight is a decreasing, piecewise linear function of time synchronized with the thrust profile above. Unlike the aircraft model the missile model is generic.

3. MISSILE/TARGET–SIMULATIONS

The simulation of a fictitious missile/target encounter is a basic module of pre–launch guidance. Beside the vehicle models and the initial states the guidance laws of both vehicles determine the outcome of the encounter. The missile is guided by Proportional Navigation. The target strategy is an approximation of optimal missile avoidance. The associated guidance law is taken from Ref. 2 and extended by additional control modes on approaching the constraints (12), (13). The simulation is stopped at a closing speed of 150 m/s. The output is the final distance R, which is taken as a measure for the kill probability. Let

$$z = (x, y, h, V, \gamma, \chi)^T$$

denote the state vector of a vehicle. Since the vehicle model and the guidance law of both missile and target are fixed R can be expressed as a function of the initial states z_M and z_T:

$$R = R(z_M, z_T) \tag{16}$$

The missile is assumed to hit if

$$R(z_M, z_T) = 1 \text{ km}. \tag{17}$$

The (arbitrary) choice of the 1 km limit shows that the final phase of the encounter, characterized by heavy target maneuvering, is exempted. We only consider the "midcourse" part of the duel. Simulation means to numerically integrate Eqs. (1)–(6) for both vehicles up to the stopping condition. The execution with a usual Runge–Kutta method advancing in small steps is termed "full simulation" in contrast to the "simplified simulation" designed for real–time application.

The simplified simulation proceeds as follows. The first two integration steps are identical with the boost and march phase of the missile. The coasting phase is partitioned in 10–sec intervals except the last one, where the stopping condition is trapped. Each step represents an initial value problem, which is approximately solved as follows. $\gamma(t)$ and $\chi(t)$ are modeled as linear functions along the subinterval. The (piecewise constant) rates of γ and χ are set in accordance with the vehicle strategies. The right–hand side of Eq. (4) is linearized about the initial altitude and speed at the beginning of the step. Thus, an ODE–system results, which can be solved in closed form. Details are given in Appendix A.

The full and simplified simulation procedures are compared for the following scenario:

		missile	target
x_0	[km]	0	30
y_0	[km]	0	0
h_0	[km]	10	4.9
V_0	[m/s]	547.8	310
γ_0	[deg]	0	–5.7
χ_0	[deg]	22.9	85.9

The final times of the full and simplified simulation are 39.5 sec and 41.2 sec, respectively. The miss distances are 6.9 km and 5.2 km. The computer time is reduced about a factor 40 by the simplified procedure, if the full simulation is performed with a fourth order Runge–Kutta method and a stepsize of 0.25 sec. The difference in final distance is the price for the speedup. In view of the uncertainties present in reality (for instance the actual target strategy) the loss of accuracy on simplifying the simulation is a minor effect.

The turning maneuvers are different in the full and the simplified simulation (Fig. 1a). In the former one turning is much sharper. Optimal missile avoidance basically consists of a dive (Fig. 1b). After a few seconds the dive is stopped since the target approaches its dynamic pressure boundary (13). The altitude histories are well resembled by the simplified simulation. The same holds for the speed profiles, where the different missile burn phases can be recognized (Fig. 1c).

4. THE RATIO CRITERION

A pre–launch guidance is designed using the miss distance (16) of a fictitious missile launch as a measure for missile performance. The guidance method applies to a combat situation with two aircraft A1, A2, each of which wants to shoot down his opponent. Let z_i denote the current state vector of Ai and

$$R_{ij} = R(z_i, z_j) \qquad (18)$$

the miss distance of a fictitious missile launch of Ai against Aj. Because of his aggressive intention each aircraft Ai approaches a firing position against his opponent Aj. In terms of the R_{ij} this means $dR_{ij}/dt<0$ as shown in Fig. 2. The solid line in Fig. 2 is the preferred outcome of A1, characterized by $R_{12} = 1$ km and $R_{21} > 1$ km at the end. The dashed line leads to a termination desired by A2.

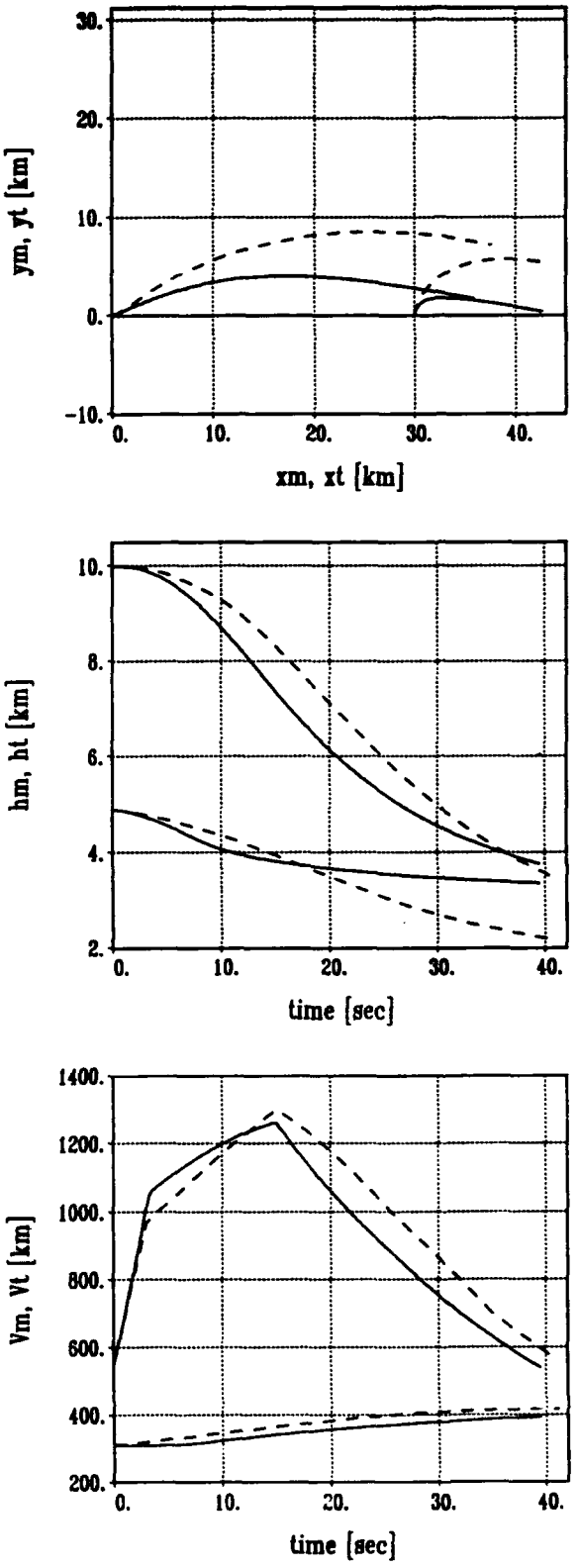


Fig. 1: Comparison of a full (solid line) and simplified (dashed line) missile/target simulation. 1a: horizontal projection. 1b: altitude histories. 1c: speed profiles.

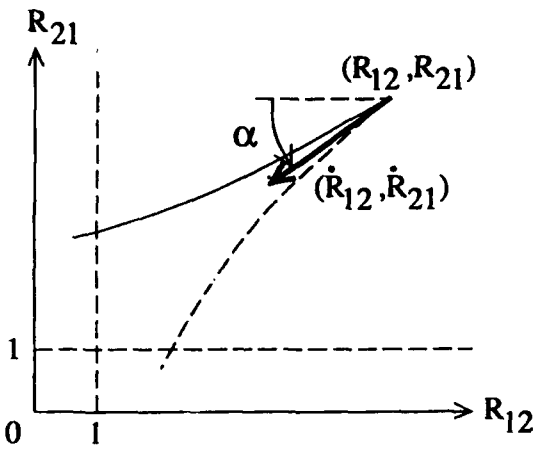


Fig. 2: One–versus–one air combat monitored in the (R_{12}, R_{21})–space.

Obviously, a reasonable strategy for A1 is to make α as small as possible. Because of

$$\tan\alpha = \frac{\dot{R}_{21}}{\dot{R}_{12}} \tag{19}$$

minimizing α is equivalent to the condition

$$\underset{n_1,\mu_1,\xi_1}{\text{minimize}} \ \frac{\dot{R}_{21}}{\dot{R}_{12}} . \tag{20}$$

According to the chain rule we have

$$\dot{R}_{ij} = \frac{\partial R_{ij}}{\partial z_1}\dot{z}_1 + \frac{\partial R_{ij}}{\partial z_2}\dot{z}_2 . \tag{21}$$

The controls n_i, μ_i, ξ_i explicitly appear in $\dot{z}_i$ if the right–hand sides of (1)–(6) are substituted. The gradients of R_{ij} with respect to z_1 and z_2 are approximated by numerical differentiation. This is the most expensive and the most difficult part of the guidance law. The difficulty arises from the necessity that $R(z_1, z_2)$ is actually differentiable with respect to z_1 and z_2. This requires special care on the design of the vehicle strategies used on evaluating R. Furthermore, it demands highest possible accuracy on trapping the terminal condition of the missile/target simulation.

The objective function (19) tends to $-\infty$ for

$$\dot{R}_{12} \to 0+0 \ \text{ and } \ \dot{R}_{21} < 0.$$

Therefore, (20) only makes sense together with the constraint

$$\dot{R}_{12} \le -\epsilon < 0 . \tag{22}$$

(22) means that A1 approaches a firing position towards A2 with a minimum "speed" ϵ. The "ratio criterion" (20), (22) can successfully be simulated on the computer. A1 actually reaches a firing opportunity first under symmetric initial conditions. However, the objective function (19) also involves the controls of A2, which are unknown to A1 in reality. Therefore, (20) is modified as follows: Consider (19) for constant γ_i and χ_i, i.e.

$$\dot{\gamma}_i = \dot{\chi}_i = 0 \implies n_i = \cos\gamma_i , \ \mu_i = 0 \tag{23}$$

for $i = 1, 2$. Then, (19), (22) only depend on the state variables and the thrust controls ξ_i. Instead of optimizing the controls n_1, μ_1 we look for the triplet $(\gamma_1, \chi_1, \xi_1)$, which minimizes (19) under the constraint (22) and the assumption $\xi_2 = 1$:

$$\underset{\gamma_1,\chi_1,\xi_1}{\text{minimize}} \ \frac{\overset{0}{R}_{21}}{\overset{0}{R}_{12}} \tag{24}$$

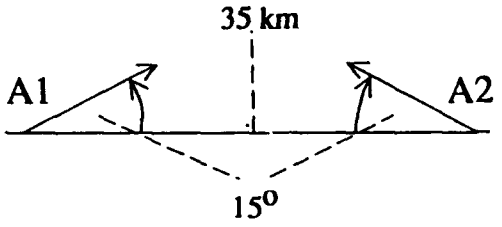
subject to

$$\overset{0}{R}_{12} \le -\epsilon < 0$$

By an "o" instead of a dot we denote the rate of R_{ij} evaluated under condition (23). The minimizing values γ^r and χ^r serve as "reference values" for the lift control: n_1 and μ_1 are such that $\gamma_1 \to \gamma^r$ and $\chi_1 \to \chi^r$.

Strictly speaking, also the gradients of R_{ij} with respect to z_1 and z_2 (see Eq. (21)) depend on γ_1 and χ_1. This dependence is neglected to limit the computational expense. Both vectors are evaluated for the actual γ_1 and χ_1 and remain unchanged on executing the minimization.

Fig. 3 shows the simulation result for the symmetric initial scenario drawn below.



Both aircraft start at $h = 7$ km, $V = 437$ m/s, $\gamma = 0$. A1 is guided by (24). By the way, the original control law as formulated in (20)+(22) leads to similar results. The R_{ij} and their gradients with respect to z_1, z_2 (see Eq. (21)) are updated twice a second only. This is the reason for the step–like histories of R_{ij} and other quantities. A2 is guided by Proportional Navigation to simulate some conventional attack guidance.

Due to the symmetry in the initial conditions R_{12} and R_{21} agree at $t = 0$. With the help of (24) A1 can build up an advantage $R_{21} - R_{12} > 4$ km. This is accomplished in less than 10 seconds; the effect of (24) is of short term nature, obviously. Thus, it does not make sense to consider longer flight times without changing A2's strategy.

Let us look more closely at the final state (Fig. 3, right picture on the top). On a missile launch by A2 A1 would escape to the left in line–of–sight direction. This would require about 90^0 turning by A1. A1's missile directed towards A2 would have to cover about the same heading difference. A1's success suggests the following interpretation. A certain vicinity to an escape direction is better than to fire the missile directly towards the opponent. A2 does the very opposite and loses. A1's missile reaches him because he must turn about 180^0 to evade. On the other hand A2's missile does not hit A1 although it could fly more or less directly towards A1. The reason is that A1 is closer to an escape direction than A2.

Now the initial condition is changed in favour of A2: $h_2 = 11$ km. The advantage is turned around by the "ratio criterion" in the first few seconds (see Fig. 4); A1 wins in spite of a speed loss of nearly 40 m/s. Obviously, a speed advantage need not be an advantage in air combat. Lower speed allows for a sharper turn to escape. This seems to be more

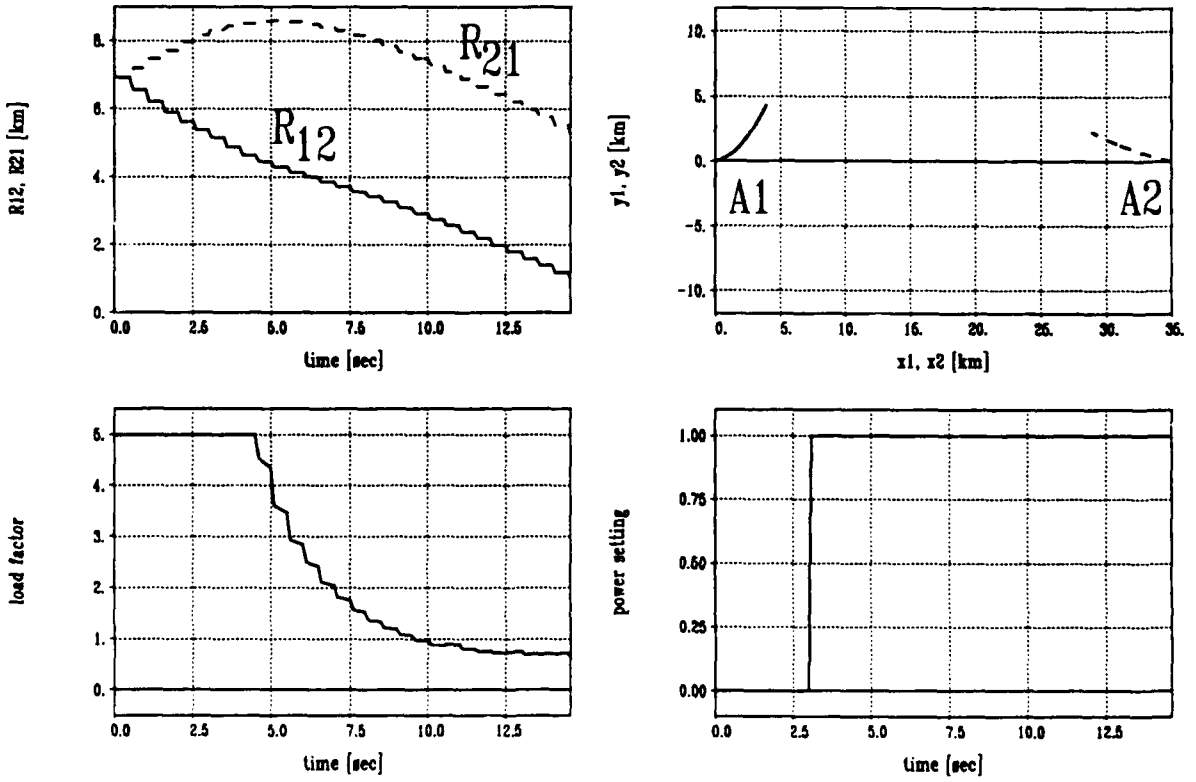


Fig. 3: Simulation with the modified ratio criterion for symmetric initial conditions.

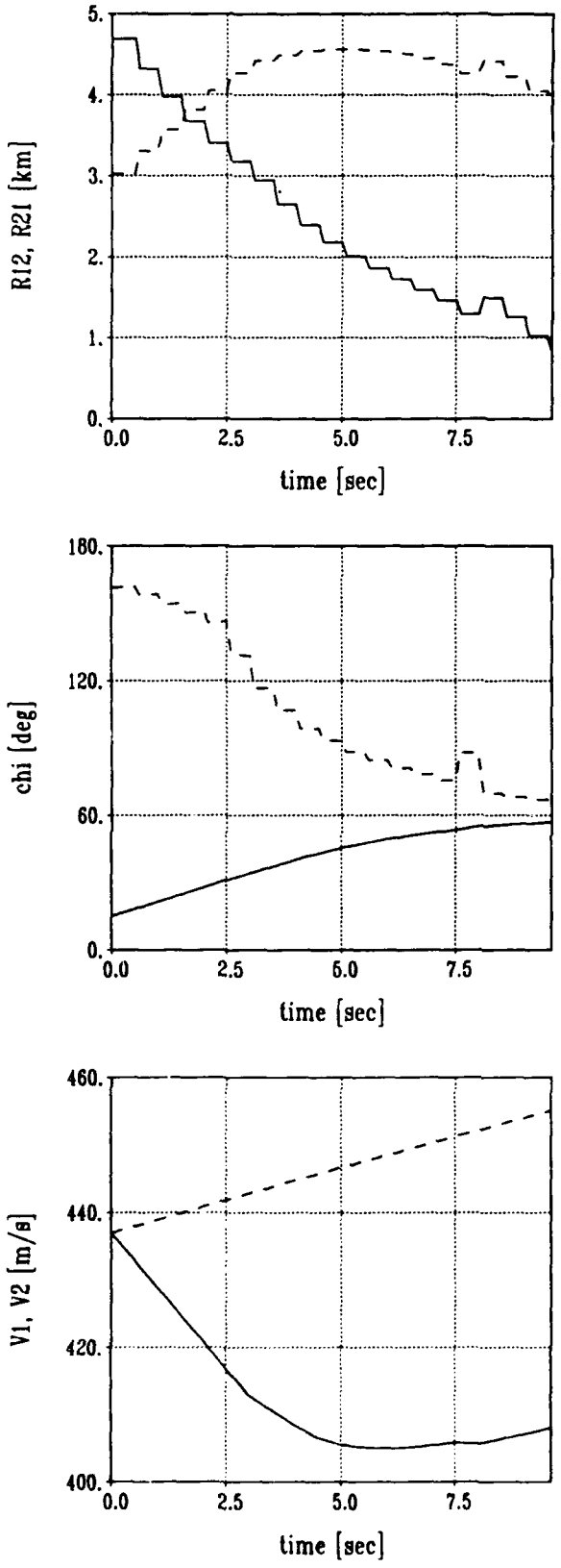


Fig. 4: Simulation with the modified ratio criterion: altitude advantage for A2.

important than to provide a high initial speed for the missile – another message conveyed by Figs. 3, 4. This is also the interpretation of the minimum thrust phase in the first seconds. In the χ–diagram the dashed line indicates the reference angle χ^r, which denotes the solution of (24) together with γ^r, ξ^r. The actual heading χ_1 tends to angle χ^r as intended by the guidance law. After t = 7.5 sec R_{12} increases slightly. This is not a contradiction to the constraint of (24) (we set ε = 200 m/s). Note that the modified version of the guidance law considers the rate of R_{ij} for constant γ and χ. The actual rate may differ significantly.

5. MINIMUM TIME PRE–LAUNCH TRAJECTORIES

The guidance method presented in this section is designed such that a firing position against a nonmaneuvering target is reached in minimum time. According to (17) the terminal condition is

$$R(z_f, z_{Tf}) = 1 \text{ km}, \tag{25}$$

where z and z_T denote the state vector of interceptor and target, respectively. The optimal control problem underlying the design is to

$$\begin{aligned} &\text{find } n(t), \mu(t), \xi(t), \text{ which} \\ &\text{minimize } t_f \text{ subject to} \\ &R(z_f, z_{Tf}) = 1 \text{ km}. \end{aligned} \tag{26}$$

A "simplified missile/target simulation" (section 3) is performed to evaluate R.

The guidance law internally approximates the optimal pre–launch trajectory starting at the current state. The approximation is done as follows. The dynamical model is reduced to Eqs. (1)–(4) the controls being γ, χ and ξ. The original controls n and μ are formally eliminated by

$$\dot\gamma = \dot\chi = 0 \implies n = \cos\gamma, \ \mu = 0 .$$

The resulting equations of motion represent a "reduced model" in the sense of Singular Perturbation Theory (Ref. 9). The reduction is such that the state variables γ and χ take the role of control variables.

The "reduced problem" is to

$$\text{find } \gamma(t), \chi(t), \xi(t), \text{ which minimize } t_f \text{ subject to } R(z_f, z_{Tf}) = 1 \text{ km}. \quad (27)$$

Let $\gamma^{\pi}(t)$, $\chi^{\pi}(t)$, $\xi^{\pi}(t)$ be the solution of (27) ("reduced solution"). The controls n and μ are such that $\gamma \to \gamma^{\pi}(0)$ and $\chi \to \chi^{\pi}(0)$. Thrust is controlled by $\xi = \xi^{\pi}(0)$.

The numerical approach for (27) is to model γ, χ and ξ as piecewise constant functions. Thus, the optimal control problem (27) is converted into a nonlinear program with the objective function t_f and the constraint (25). The parameters are the constant γ–, χ– and ξ–values in the subintervals and the unknown flight time. The iterative solution process is performed by subroutine SLSQP (Ref. 10).

On each iteration the optimization method within the guidance algorithm requires one ore more solutions of the (reduced) equations of motion. Since standard integration methods are not suited for real–time application, a special integration procedure for the the underlying dynamical model has been designed. Details are given in Appendix B.

The guidance method outlined above is tested in "simulations": The dynamic system (1)–(6) is integrated forward with the controls issued by the guidance method. The simulation is the "outer loop" in contrast to the "inner loop" represented by the guidance law. An overview is given by the following scheme.

simulation: Integrate the equations of motion with the controls taken from the

guidance law.

1. Solve (27) with the initial state being the current state. Result: "reduced solution" serving as reference trajectory.
2. Select n and μ such that the reference trajectory is tracked.

The reduced solution (step 1) is updated each second only since it varies slowly. Step 2 is executed on each call of the guidance algorithm using the latest update of the reduced solution.

In the simulation (outer loop) the target flies along a straight line at constant altitude with constant speed. Thus, the "actual" target flight path agrees with the internal prediction of the guidance algorithm (inner loop). The simulation is stopped as soon as terminal condition (25) is satisfied.

The initial conditions for the following example are given below:

		inter–ceptor	target
x_0	[km]	0	30
y_0	[km]	0	0
h_0	[km]	1	5
V_0	[m/s]	234.4	200
γ_0	[deg]	0	0
χ_0	[deg]	120	29

The interceptor begins with a turning maneuver to approach the target. This phase is characterized by high load factor and $\gamma < 0$ (see Fig. 5). The rest of the trajectory is a climb taking place in a vertical plane more or less. Thus, the constraints (12) and (13) do not become active, although they could easily be incorporated into the guidance method (see Ref. 8). After an intermediate phase with small γ there is a pronounced climb portion at the end terminating with $\gamma_f = 32^0$. The physical explanation is that missile range is increased at high altitude due to small air density. One can imagine that this effect does not show up with simple "firing envelopes". The final values γ_f in this study are larger than 30^0 throughout. This is in accordance with Ref. 11 but is a remarkable difference to Ref. 7, where γ_f is mainly less than 10^0. Fig. 5 also shows the target switching from idle mode to missile avoidance strategy at launch time. In the post–launch phase the target accelerates with maximum thrust and dives until it reaches the dynamic pressure limit (13).

In Fig. 5 the simulation with the feedback guidance is compared to the optimal solution of problem (26), obtained with conventional nonlinear programming techniques. The features described above are even more marked at the optimal solution. The associated flight time – the minimum attainable in this scenario – is 95.93 sec. With a flight time of 97.69 sec the simulation is only slightly longer. The similarity of simulated and optimal altitude and speed histories confirms the near–optimality of the guidance method. One evaluation of the guidance command including both steps (step 1 is evaluated once in a second of flight time only) takes at most a third of a second computer time on the average on an IBM 3090 main frame.

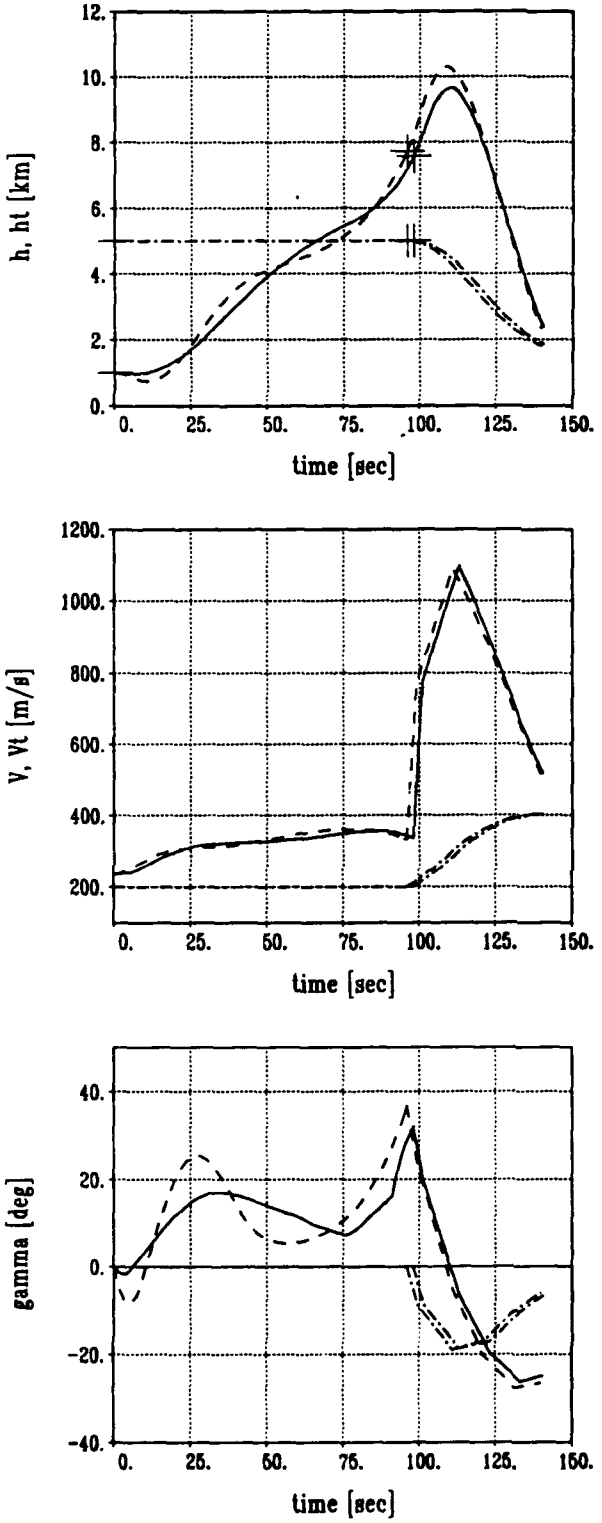


Fig. 5: Time–optimal pre–launch maneuver. Optimal solution (dashed line) and simulation with a near–optimal feedback guidance (solid line). Dash–dotted line: target state. Beyond the launch times (crosses in the uppermost diagram) the interceptor trajectories are continued by the missile state.

Fig. 6 shows the influence of the target post–launch maneuver on the interceptor's pre-launch trajectory. As an alternative to the missile avoidance strategy used so far the target now turns to line–of–sight direction at constant altitude and speed. Because of the initial conditions there is nearly no turning in this example. Clearly, the target can now be hit earlier from a longer distance. A firing position is already reached after 42.85 sec (compared to 97.69 sec, if the target tries to outrun the missile). γ monotonously increases up to a maximum of 44^0 at launch time.

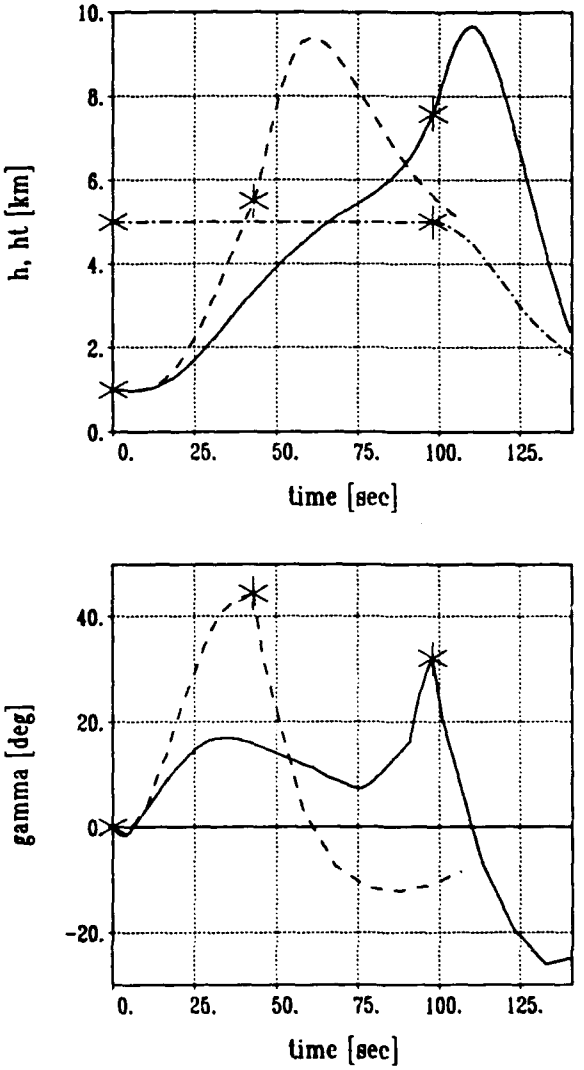


Fig. 6: Time–optimal pre–launch trajectories for different target behaviour in the post–launch phase. Solid line: The target tries to avoid the missile. Dashed line: The target flies at constant altitude and speed. Beyond the launch times (asterisks) the interceptor trajectory is continued by the missile state. Dash–dotted line: target state for the case of active missile avoidance.

Next, the effect on altering the missile guidance law is examined. An optimal pre–launch trajectory terminates at high altitude with a large flight path angle to increase missile range. This lofting effect is immediately destroyed by the missile descending to the lower target altitude. To eliminate this paradox behaviour the missile is given reference values γ_b^r and γ_m^r for the flight path angle in the boost and march phase. γ_b^r and γ_m^r are optimized in the guidance algorithm within certain bounds. Three cases are considered:

a) The missile guidance is unchanged compared to previous examples.
b) $\gamma_b^r, \gamma_m^r \leq 11^0$
c) $\gamma_b^r, \gamma_m^r \leq 45^0$

The initial values for this experiment are the following:

		inter–ceptor	target
x_0	[km]	0	50
y_0	[km]	0	0
h_0	[km]	5	5
V_0	[m/s]	400	200
γ_0	[deg]	0	0
χ_0	[deg]	0	0

The table below gives the interceptor's flight time t_f and the missile flight time t_m estimated in a simplified missile/target simulation (the target employs missile avoidance strategy). The trajectories are depicted in Fig. 7.

	t_f [sec]	t_m [sec]
a)	90.40	57.65
b)	74.88	81.79
c)	35.66	127.94

The results confirm the expected trend. In cases b) and c) the optimal values of γ_b^r and γ_m^r are on the upper bounds. Missile range is drastically increased since the missile is allowed to climb up in the burn phase. Accordingly, a firing position is reached earlier and the missile flight time increases.

After the typical dive target altitude levels out at about 1 km. Again, this is the effect of a control mode to comply with the dynamic pressure constraint (13). The structure of the interceptor's pre–launch trajectory is similar as before. γ passes a local minimum for longer flight times (a) and b)); for short flight times (case c)) γ monotonously increases. It should be added that the interceptor is constrained by $\gamma \leq 45^0$. This constraint is active in b) and c).

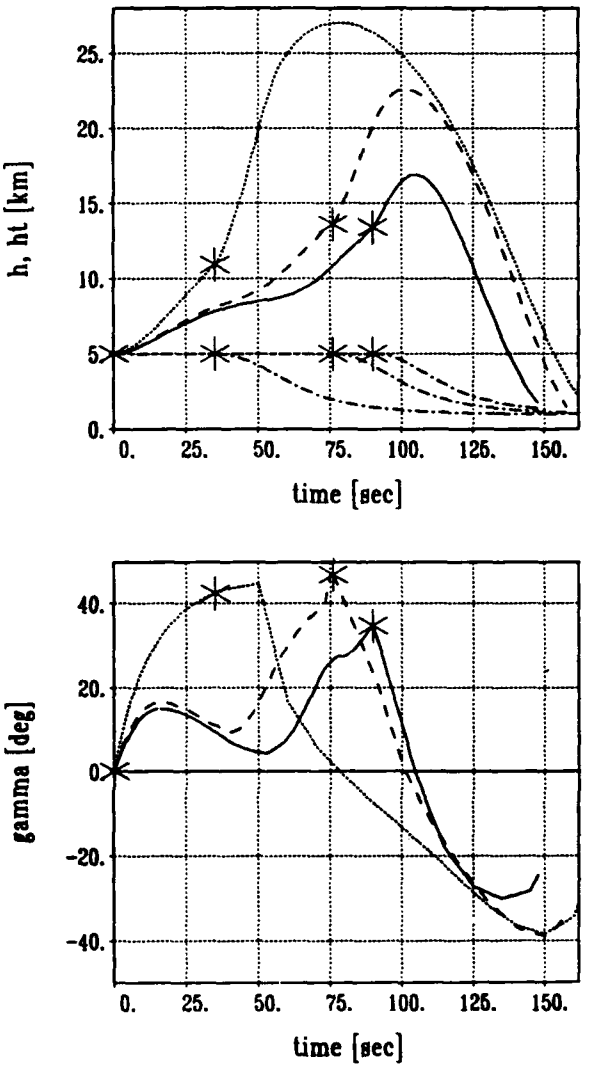


Fig. 7: Time–optimal pre–launch trajectories for different missile guidance in the burn phase.
Solid line: usual collision course guidance.
Dashed line: climb up to $\gamma = 11^0$.
Dotted line: climb up to $\gamma = 45^0$.
Dash–dotted line: target state.
Beyond the launch times (asterisks) the interceptor trajectories are continued by the missile state.

6. CONCLUSIONS

Aircraft guidance in the pre–launch phase strongly depends on effects which are not covered by simple "firing envelopes":

- The range of a missile is increased if it is launched at high altitude and elevation.
- The chance of the target to escape strongly depends on its current direction.

A missile/target simulation must be performed on–line to quantify the above effects. The simulation must be

- real–time capable and
- smooth with respect to the initial state,

since any kind of guidance will need the sensitivity of the results with respect to the initial state.

The simulation (and any guidance, which is based on the simulation)
strongly depends on

- the vehicle model and
- the strategy

of the opponent, who can alternately take the role of missile or target in the simulation. Knowledge about his model is an information problem. The assumption about his strategy as a target determines the risk a pilot is willing to take. A firing position against an optimally evading target bears the highest risk for own survival.

Based on missile/target simulations a guidance method ("ratio criterion") is presented, which successfully improves the strategic position in air combat. However, the improvement is a short term effect adequate for the immediate pre–launch phase only. Also, the guidance command need not be defined as soon as the opponent tries to escape. Simulations show that an advantageous strategic position is characterized by the ability to turn to line–of–sight direction in short time. This may require reduced speed and an angular vicinity to the escape direction even if the missile cannot be directly aimed at the opponent. Obviously, high speed need not be favourable on entering a beyond visual range combat situation.

Another guidance method leads to a firing position against a nonmaneuvering target in nearly minimum time. The firing position is characterized by high altitude and elevation at the cost of speed. The terminal phase of the pre–launch trajectory is a steep climb, which constitutes the whole trajectory, if the flight time is short. Missile range is significantly increased if the climb may be continued by the missile during its burn phase.

The guidance methods in this paper are examples how to taylor trajectory simulation and optimization for real–time application. The basic idea is to design a fast integration method for the special differential equation system. A quite different philosophy is to precompute sets of (optimal) trajectories ("extremal fields") and to pick an element fitting to the current situation. The nice feature of this approach is that it needs nearly no computing time. However, precomputed data depend

- on special vehicle models and
- the special mathematical formulation of the mission objective.

Precomputed data must be recalculated if a detail in one of the two items above is altered. Because of enhanced flexibility we prefer the on–line computation of required flight path predictions. This approach gains attraction from the rapid development of high performance computers and the effectiveness of modern numerical algorithms.

Beside partial automation of aircraft guidance the methods described in this paper could be used in a pilot's decision aid. By means of missile/target simulations the chance to hit an aggressive or defensive target is estimated. This yields a lower and an upper bound for the kill probability. The same calculation is repeated with reversed roles, i.e. missile firing by the opponent. The results – the chance to hit and the degree of threat – are continuously displayed to the pilot. While this is only an instantaneous risk analysis, fictitious minimum–time pre–launch trajectories of the own and the adversary aircraft can be compared to predict the event of an encounter. The superior fighter reaches a firing opportunity first. In any case the results would have informational character and would not override the pilot's decision.

REFERENCES

1. Menon, P.K.A. and Duke, E.L., "Time–Optimal Aircraft Pursuit–Evasion with a Weapon Envelope Constraint", Proceedings of the American Control Conference, San Diego, Ca., May 1990, pp. 2337–2342.
2. Shinar, J. and Gazit, R., "Optimal No–Escape Envelopes of Guided Missiles", AIAA Paper No. 85–1960 CP. AIAA Guidance, Navigation and Control Conference, Snowmass, Co., 1985.

3. Järmark, B., "A Missile Duel Between Two Aircraft", J. Guidance, Control & Dynamics, Vol. 8, No. 4, July–August 1985, pp. 508–513.
4. Moritz, K., Polis, R. and Well, K.H., "Pursuit–Evasion in Medium–Range Air–Combat Scenarios", Comp. Math. Appl., Vol. 13, No. 1–3, 1987, pp. 167–180.
5. Grimm, W., Prasad, U.R. and Well, K.H., "Open–Loop Guidance for Pre–Launch Maneuvering in Medium–Range Air–Combat", Proceedings of the 27th IEEE Conference on Decision & Control, Austin, Texas, Dec. 1988, pp. 1442–1447.
6. Prasad, U.R., Grimm, W. and Berger, E.G., "A Feedback Guidance for Pre–Launch Maneuvering in Medium–Range Air–Combat with Missiles", in: Differential Games and Applications, T.S. Basar and P. Bernhard (eds.), Lecture Notes in Control and Information Sciences, Vol. 119, 1988, pp. 86–96.
7. Shinar, J. and Spitzer, A., "Global Optimization of Medium–range Air–to–Air Interceptions", Proceedings of the American Control Conference, Atlanta, Georgia, May 1988, pp. 977–982.
8. Grimm, W., "A Numerical Approach for On–Line Guidance of Aircraft", Proceedings of the 25th Conference on Decision & Control, Athens, Greece, Dec. 1986, pp. 683–687.
9. Ardema, M.D., "An Introduction to Singular Perturbations in Nonlinear Optimal Control", in: Singular Perturbations in Systems and Control, M.D. Ardema (ed.), CISM Courses and Lectures No. 280, Springer, Wien – New York, 1983.
10. Kraft, D., "A Software Package for Sequential Quadratic Programming", DFVLR–FB 88–28, 1988.
11. Well, K.H., "Medium Range Intercept Maneuvers with a Missile", Proceedings of the American Control Conference, San Diego, Ca., May 23–25, 1990, pp. 2343–2348.

APPENDIX A: SIMPLIFIED INTEGRATION OF THE EQUATIONS OF MOTION – COMPLETE MODEL

Let x_0, y_0, h_0, V_0, γ_0, χ_0 denote the state vector of a vehicle at the beginning of an integration step. Within the step the rates of γ and χ are given constants:

$$\gamma(t) = \gamma_0 + t \cdot \dot{\gamma}\,, \tag{A1}$$

$$\chi(t) = \chi_0 + t \cdot \dot{\chi}\,. \tag{A2}$$

$\dot{\gamma}$ and $\dot{\chi}$ are selected in accordance with the guidance strategies of missile and target, respectively. From Eqs. (5), (6) the controls can be determined:

$$n \cos\mu = \frac{\dot{\gamma}\, V}{g} + \cos\gamma \tag{A3}$$

$$n \sin\mu = \frac{\dot{\chi}\, V}{g} \cdot \cos\gamma \tag{A4}$$

$$n = \sqrt{(n\ \cos\mu)^2 + (n\ \sin\mu)^2} \tag{A5}$$

The load factor in D(h,V,n) is replaced by (A3) – (A5). Thus, drag becomes a function of h, V, and cosγ:

$$D(h, V, n) = \bar{D}(h, V, \cos\gamma) \tag{A6}$$

Now, T and $\bar{D}$ are linearized about h_0, V_0 and γ_0. The resulting expressions contain the altitude difference $h(t) - h_0$, which is approximated by

$$h(t) - h_0 \approx V_0 \int_0^t \sin\gamma(\tau)d\tau = V_0\,(\cos\gamma_0 - \cos\gamma(t)) \,/\, \dot{\gamma} \tag{A7}$$

Finally, Eq. (4) takes the form

$$\frac{d}{dt}(V - V_0) = a \cdot (V - V_0) - A \cdot \sin\gamma - B \cdot \cos\gamma - C. \tag{A8}$$

a, A, B, C are constants (within the respective step). They mainly consist of partial derivatives of thrust and drag, which are evaluated at the initial point of the step. (A8) is a linear inhomogeneous equation, which can be solved in closed form. Having the V(t)–solution x(t), y(t), and h(t) can also be represented in closed form since $\gamma(t)$ and $\chi(t)$ are assumed to be linear.

APPENDIX B: SIMPLIFIED INTEGRATION OF THE EQUATIONS OF MOTION – REDUCED MODEL

The "reduced" dynamic model with the control variables γ, χ and ξ consists of the following differential equations:

$$\dot{x} = V \cos\gamma \cos\chi \qquad \text{(B1)}$$

$$\dot{y} = V \cos\gamma \sin\chi \qquad \text{(B2)}$$

$$\dot{h} = V \sin\gamma \qquad \text{(B3)}$$

$$\dot{V} = g\left[\frac{T - D(h,V,n)}{W} - \sin\gamma\right] \qquad \text{(B4)}$$

The load factor in the drag function is replaced by $n = \cos\gamma$. Analytical approximations of T and D are chosen which let Eq. (B4) appear in the form

$$\dot{V} = \sum_{i=0}^{2} a_i(h, \gamma, \xi) \cdot V^i. \qquad \text{(B5)}$$

E.g. a suitable model of T_{min} is

$$T_{min} = \sum_{i=0}^{2} f_i(h) \cdot M^i.$$

To solve (27) γ and ξ are modeled as piecewise constant functions. If also the h–argument in a_i is kept constant within a subinterval, (B1)–(B3) together with (B5) have a piecewise closed–form solution.

FUZZY GUIDANCE SYSTEM EVALUATION

by

J.R. Martin, F. Sanchez, P.V. Cuenca, L.M. Rodriguez and J.B. Ecija
Laboratorio de Guiado
Instituto Nacional de Tecnica
Aerospacial
Carretera de Ajalvir, Km 4
Torrejon de Ardoz
Madrid 28850
Spain

ABSTRACT

A study is described that compares the capability of a fuzzy logic controller with a classical controller P+D (proportional plus derivative). The model used for this investigation is the attitude control of a microsatellite launcher, during the first stage of flight.

This model has been chosen because performs a very unstable plant, the launcher is understood without stabilization surfaces and only the body is considered as generator of aerodynamics forces. Movable-nozzle TVC (Thrust Vector Control) will be used as flight control device. The problem is studied under 3DOF (three degrees of freedom) simulation at two levels, software and hardware. The concept formulated in this work is the analysis of the fuzzy rules, that performs robust control during the flight.

It has been used as perturbations, a wind profile and missalignments in the launcher. The wind model generates a wind velocity vector, which is constant throughout each run. The wind velocity vector is parallel to the surface of the earth.

INTRODUCTION

The study has two levels:
-Simulation under software model of different subsystems. A FORTRAN77 code subroutines are studied, in order to generate the rules needed in the fuzzy logic controller emulator. Rate gyro model, electronic integration, movable nozzle actuator are modelled with real test data.
-Complete simulation with HWIL (Hardware In the Loop) testing, combines a real time simulation with flight hardware. This hardware consist:

-Digital real time Gould 32/97 computer.

-Interfaces to the digital computer (A/D, D/A).

-Fuzzy logic Hardware.

-Rate gyro and electronic integrator.

-One rotational motion simulator (CARCO flight table).

-Torque simulator (CARCO)

Finally software and hardware models are compared under Montecarlo method an conclusions are presented.

LAUNCHER MODEL

The basic vehicle configuration is presented in figure 1. Some of the variables used in this report are defined in this figure, the list of simbols is

a distance from gimbal point to center of gravity,m
b distance from center of gravity to center of pressure, m
c distance from nose to center of pressure,m
CG center of gravity
CP center of pressure
GP gimbal point
g gravitational acceleration, m/sec^2

I moment of inertia, Kg-m^2
K_θ attitude gain constant
$K_{\theta'}$ attitude rate gain constant
m mass, Kg
s Laplace operator, sec^{-1}
t time, sec
T thrust, N
v velocity, m/sec
α vehicle angle of attack, rad
α_w wind angle, rad
δ thrust vector deflection angle, rad
θ launcher attitude, rad
θ' launcher rate attitude, rad/sec
θ_c commanded launcher attitude, rad
q dynamic pressure, N/m^2
S area of maximum body section, m^2
M Mach number
σ air density, Kg/m^3
C_d drag force coefficient
$C_{n\alpha}$ normal force coefficient

Figure 1 shows the vehicle aerodynamic forces that act on it. These forces may be assumed to be concentrated at a single point, the vehicle center of pressure. In this case results an aerodynamically unstable vehicle, that is stabilized by the control system. A rigid body configuration is assumed, and three degrees of freedom model is considered. Therefore, vehicle bending, and aeroelastic effects are not taking in account. The feedback variables are the attitude and attitude rate.

The vehicle equations of motion for a rigid body configuration are:

$$m\dot{v} = T\cos(\theta-\gamma-\delta) - D\cos(\theta-\gamma)$$

$$-N\sin(\theta-\gamma) - mg\sin\gamma$$

$$mv\dot{\gamma} = T\sin(\theta-\gamma-\delta) - D\sin(\theta-\gamma)$$

$$+N\cos(\theta-\gamma) - mg\cos\gamma$$

$$I\ddot{\theta} = Ta\sin\delta + Nb$$

$$\alpha = \theta - \gamma - \alpha_w$$

where

$$q = 1/2\ \sigma\ v^2$$

$$D = q\ S\ C_d$$

$$N = q\ S\ C_{n\alpha}\ \alpha$$

$$C_d = C_{d0} + C_{n\alpha}\ \alpha^2$$

The vehicle parameters and aerodynamic data are those of a hypothetical launch vehicle with the following initial parameters:

-total length.....20 m
-first stage length.....10 m
-nose length.....1 m
-diameter.....1 m
-distance from gimbal point to center of gravity...8.2 m
-distance from center of gravity to center of pressure.....11.2 m
-thrust at sea level.606500 N
-time of flight.....44 s
-mass.....19421 Kg

Figure 2 shows the value of the C_{d0} coefficient as a function of Mach number that is the same for all altitudes. The $C_{n\alpha}$ coefficient is considered constant, and equal 2.

ACTUATOR MODEL

TVC are used as control device to stabilize the system, and is performed by a movable nozzle through an hydraulic actuator. The transfer function between the δ_c (delta command) and the real deflection of the thrust vector δ is a second order transfer function approximation, obtained from a real test data of the TVC actuator, with damping coefficient 1.4 and natural frequency 35 rad/s, then

$$\frac{\delta}{\delta_c} = \frac{1}{A\,s^2 + B\,s + 1}$$

with

$A = 8.16\ 10^{-4}\ sec^2$
$B = 8.0\ 10^{-2}\ sec$

The maximum deflection angle permitted to maintain stability when disturbances are presented is 3

degrees.The hardware model is performed by a hydraulic torque table, that has a delay in order to simulate the transfer function.

GYRO MODEL

the measurement of the rate and attitude angle is performed by a rate integrating gyro NORTHROP G1-G6-650B, that is modeled by a second order transfer function approximation, with damping coefficient 0.625 and natural frequency 180 rad/s.

The transfer function employed in the software model is

$$\frac{\theta}{\theta_i} = \frac{1}{C\,s^2 + D\,s + 1}$$

with

$$C = 3.0\ 10^{-5}\ sec^2$$
$$D = 6.87\ 10^{-3}\ sec$$

One axis of the flight table, is used to mount the RIG when the hardware model runs.One of the five degrees of freedom can be feedback with rate and will be used to move the table in this mode.

SIMULATION OVERVIEW

Complete software model is introduced in the GOULD computer that will run out of time, because the routine that performs the fuzzy rules takes too time, that's no problem because there are not hardware in the loop.Then the results can be compared with those obtained when the simulation runs in real time with hardware in the loop.Figure 5 shows a schematic block diagram that highlight the interconnection between the different components of the software simulation.

Fourth order Runge-Kutta integrator are used to obtain the numerical integration of the differential equations that performs the dynamic of the launcher, actuator model, gyro model and electronic integration.In order to decide on the sampling frequency in the simulation, we have performed several runs with the dynamic of the launcher and taking in account the length of the bus word, the stability of the integrator, the bandwith of other components of the simulation and the error propagation, the sampling frequency will be 200 Hz. This frequency doesn't permit to run the simulation in real time when the fuzzy emulator is running in the computer.

Figure 6 shows the block diagram of the hardware in the loop simulation, that consist in replacing the model of the RIG for the real RIG and the flight table, the actuator for the torque table and the fuzzy emulator for the hardware prototype that will be described later.

The classical and the fuzzy controller are compared in both simulations hardware and software.

ERROR SOURCE

The launch attitude is the only error considered and have a random variation between +1 degree and -1 degree on the initial alignment $\theta_0 = 90$ degrees. A constant probability density function is employed to obtain the random misalignement.

WINDS

Figure 4 shows the wind model profile that generates a wind velocity vector that is parallel to the surface of the earth.This profile will be the same in all runs without variations.

The inclusion of wind require a change in the way the angle of attack is calculated in the simulation.

AUTOPILOT

Figure 3 shows the autopilot block diagram that is studied to obtain the classical controller as a reference to compare with the fuzzy controller.The autopilot is characterized by the gains K_θ and $K_{\dot\theta}$, that are calculated in order to obtain a good control during the first stage of flight and will be constant all the time.Pole placement method is employed to obtain a robust control during the flight. The analysis is performed by linearizing about the vehicle equations of motion at constant

time.This procedure results in a set of third order equations an the autopilot gains constants are calculated in a set of times during the flight(one of those are for the maximum dynamic pressure). The classical autopilot will run in the GOULD computer for all the cases under study. The requirements needed to obtain the gains, are

-the nominal attitude θ_n = 90 degrees
-maximum deviation of θ_n 1 degree
-maximum attitude rate θ'= 1 degree/sec

With this requirements we obtain that the gains are

$K_\theta = 2 \qquad K_{\theta'} = 1$

DESIGN OF THE FUZZY LOGIC CONTROLLER

The fuzzy controller presented performs the following type of fuzzy inferences

if x is Xi and y is Yi, then c is Ci

Where "i" indicates the rule number.

Each control rule has two antecedents and one consequent.The linguistic elements used that are named labels are

PL	positive large	(+5.0)
PM	positive medium	(+3.3)
PS	positive small	(+1.7)
ZR	approximately zero	(0.0)
NS	negative small	(-1.7)
NM	negative medium	(-3.3)
NL	negative large	(-5.0)
NG	negation	(not defined)

and are defined by their membership function that will be triangle-shaped,the center point of each label is the number beside the label that corresponds to volts in the hardware.

The set operations of fuzzy sets will be defined via their membership functions,and Min-Max operations are utilized to perform the rules (1). Figure 7 shows an example of how works the fuzzy controller with two rules.The fuzzy emulator has to reproduce the characteristics of the hardware that can perform 12 rules only, then the maximum rules permitted in the emulation are 12 rules.

To obtain the rules that control the launcher under the limitations imposed it would be necessary an expert to select the suitable rules that could control the launcher for all situations that could be presented.In our case we take as reference the classical autopilot and changing the initial conditions we will obtain the global tendency of the system for this type of errors.

Figure 16 shows the state space attitude and rate of the nominal run with the initial conditions θn = 90 degrees,θ'=0 degrees/sec and figures 17 and 18 the runs with initial conditions, θ=89 degrees and θ=91 degrees with θ'=0 degrees/sec respectively, under the software model and the wind profile.The region of interest of the state space is taken to be from -1 degrees/sec to +1 degrees/sec for the rate and from 89 degrees to 91 degrees for the attitude.A cell state space is constructed to match the results obtained under different initial conditions from the classical model into the fuzzy labels, then we obtain the different possible paths that performs the domain of interest and the zones where is necessary the existence of rules. The frequency of each cell indicates the necessity of more that one rule and obtain the overlapping of the rules to get more effective inferences.

The results obtained from the figures 16,17 and 18 are matched in the cell state space of the figures 8,9 and 10 respectively. Is interesting highlight that exist discontinuity between cells but is due because the sampler frequency is not sufficient to get points into inter cells and the high speed of the classical control to go into the periodic zone. The periodic zone is determined by the set of cells that have the highest frequency. If the wind is blowing in the opposite direction, the results are symmetric and is considered when the influence zone is determined.Figure 11 shows the interest zone and figure 12 shows the periodic zone. With this results

the figure 14 shows a set of seven rules that were programed but the launcher turned unstable at 22 sec of flight .That means that the influence zone is bigger than that obtained before ,(see figure 13) then a set of eleven rules were programed (see figure 15) with satisfactory results . The figure 19 shows the output attitude and rate when the fuzzy controller emulator is programed with eleven rules, the results seem to be satisfactory until 33 secónds after launching, after this time the launcher is maintained stabilized . Figure 20 plots the attitude as a function of time but the rate presents oscillations with values 1.5 degrees/sec. Adding two rules more figure 21 shows that the amplitude of the oscillation falls inside the values permitted. Figures 22 and 23 plot the attitude and rate as function of time with thirteen rules.

HARDWARE PROTOTYPE

The fuzzy controller hardware developed by Takeshi Yamakawa, are defined in (1), is a prototype that can performs 12 rules in parallel running a high-speed. Each rule has three antecedents and one consequent that can be assigned to different types of membership function shapes, in this case we assign triangle-shapes in all cases, and the center of gravity method is adopted to transform the fuzzy output in non fuzzy output. The voltage range adopted by this prototype is between -5 volts and +5 volts and we have to adapt this range to the others hardware systems involved in the simulation. More detailed information about how works this prototype can be seen in (2).

SOFTWARE SIMULATION

Figure 5 shows the block diagram of the software simulation employed to compare the fuzzy controller and the classical controller. Montecarlo method has been used to compare the behavior of both controllers at the end of first stage at 44 sec of flight time. We will focus to the attitude at this time, when random initial misalignement is randomly varied. Figure 24 shows the attitude as function of time without missalignment (only with wind profile) of both controllers, the results are inside the basked but the behavior of the classical controller seems to be better. For the same case figure 25, 26,27 show the rate attitude as function of time, the thrust deflection angle as function of time and the state space attitude and rate respectively.

The results of 200 runs under Montecarlo method are presented by the distribution function of the figure 28, where the classical controller has better results. Is important to take in account that this don't means that the classical controller are absolutely better, because the fuzzy controller only has 11 rules and the deffuzzyfier could be changed to obtain better results.

HARDWARE MODEL

The block diagram of the hardware model is shown in figure 6, in this case the fuzzy controller is the prototype that is in the loop of the simulation. The results are slightly the same as the software model, the difference is presented by the noise involved in the simulation. The attitude ,the rate attitude and the trust deflection angle as function of time, are presented in figures 29,30 and 31, respectively.

CONCLUSIONS

The problem formulated here is how is the behavior of fuzzy controllers in the aeronautics field, and if is interesting go on in the study of this types of controllers, we are not interested in optimize a guidance system for a specific launcher in this study. In this way the conclusions are satisfactory. The fuzzy controllers seems to have more robustness than the classical controllers, but the analysis of this type of controllers fails in the absence of a closed theory. There are several methods like the cell to cell method of (3). The method employed in the analysis of the fuzzy rules has been done with the classical controller, but the same

method can be employed from the unstable dynamic model analyzing smaller cells by the same method in the time scale of one step of integration.

REFERENCES

1. Takeshi Yamakawa, "Stabilization of an inverted pendulum by a high-speed fuzzy logic controller hardware system". Fuzzy Sets and Systes 32(1989) 161-180 (North-Holland).

2. Takeshi Yamakawa, "High-Speed Fuzzy Controller Hardware System: The Mega-FIPS Machine".

3. Yung-Yaw Chen and Tsu-Chin Tsao,"A New Approach for the Global Analysis of Fuzzy Dynamical Systems". Proceedings of the 27th Conference on Decision and Control. Austin Texas, December 1988.

4. Takeshi Yamakawa, "Fuzzy Microprocessors -How it Works-". Department of EE/CS. Kumamoto University.

5. Report nº. t.r.808. Bristol aerojet Limited.Banwel,December 1976.

6. INTA report nº I-221/430/89.003

7. R. Akiba, H. Ionara. "Development of movable nozzle for solid rocked motor". American Institute of Aeronautics and Astronautics, 1983.

8. Juergen E. Ackermann,"A Robust Control System Design".Pro.Joint Automatic Control Conference,Denver,June 17-20,1979.

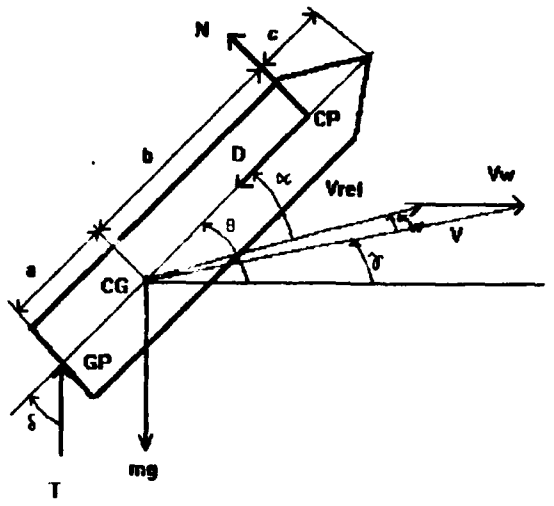


FIGURE 1 .-Launcher model

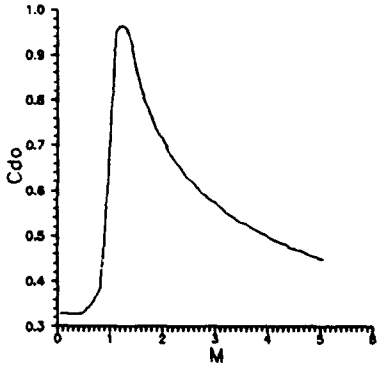


FIGURE 2

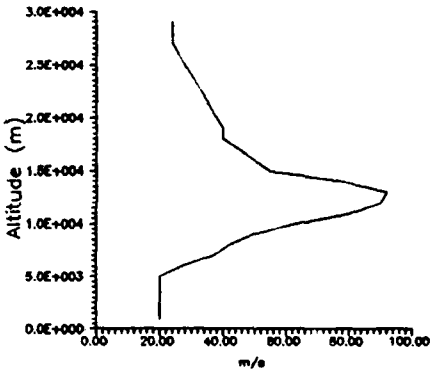


FIGURE 4.- Wind velocity as function of altitude

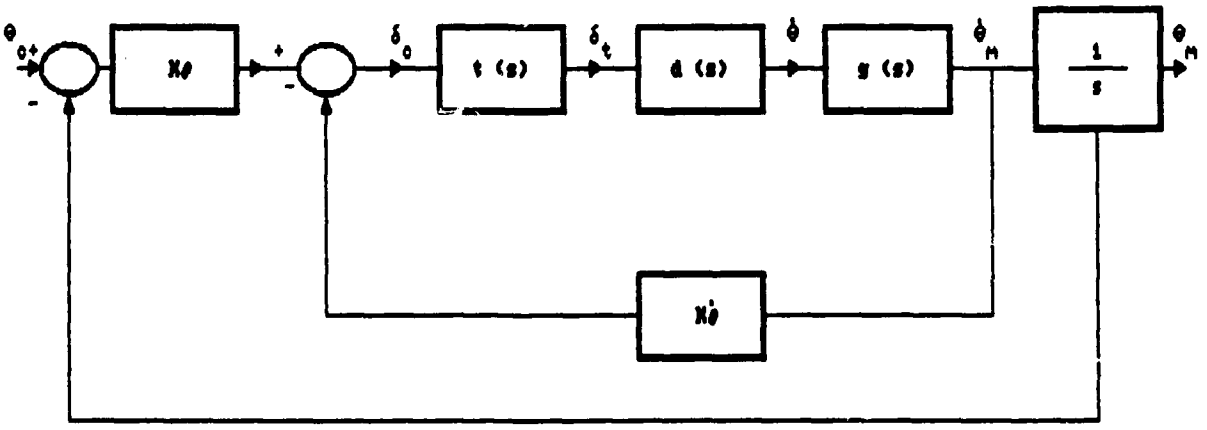


FIGURE 3.-Classical autopilot

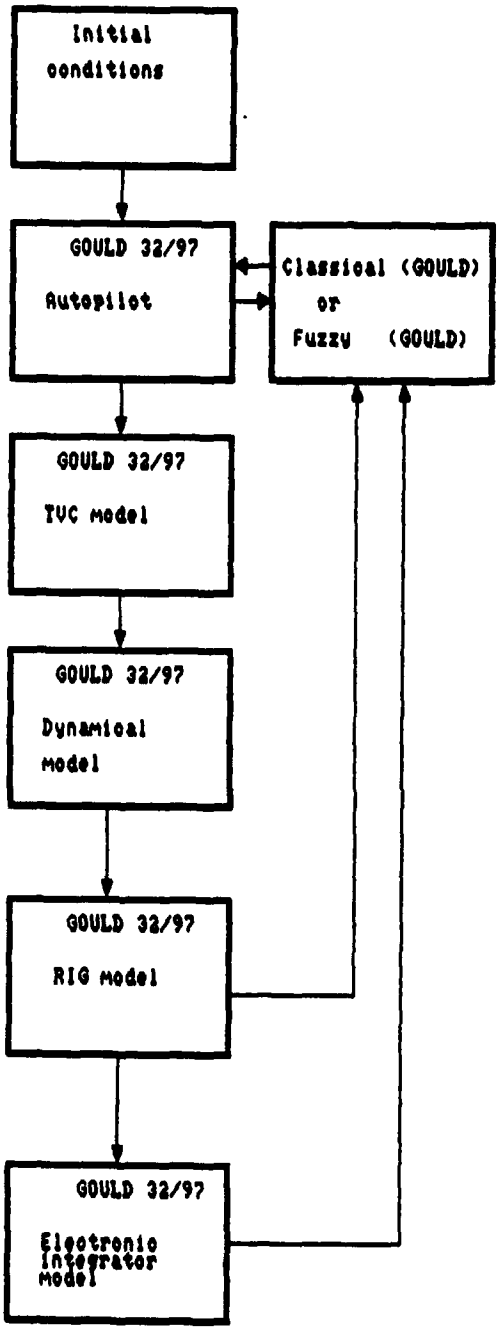


FIGURE 5.-Software block diagram

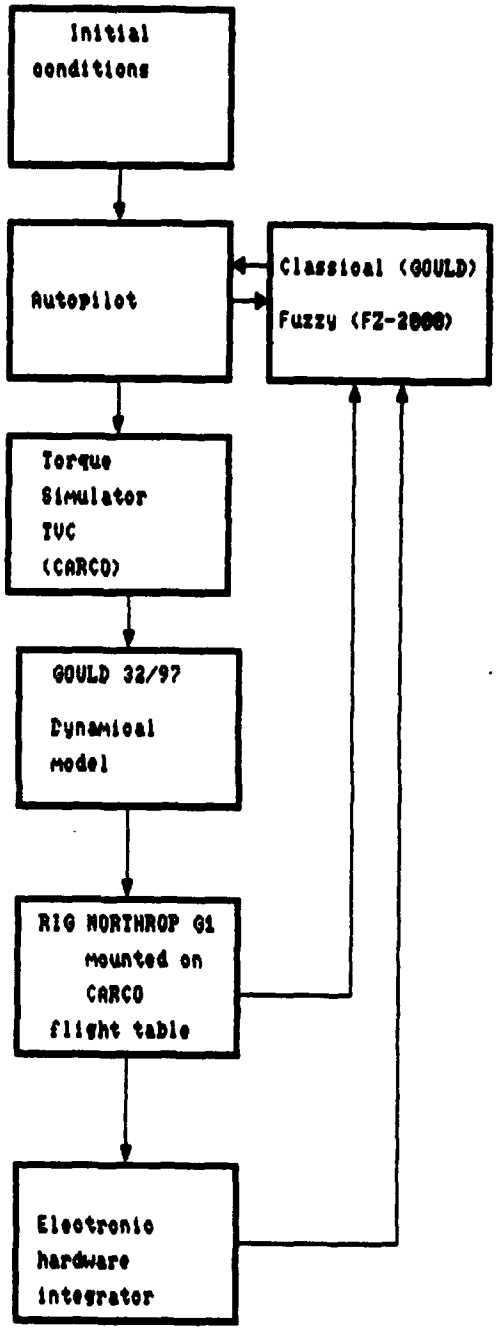


FIGURE 6.-Hardware block diagram

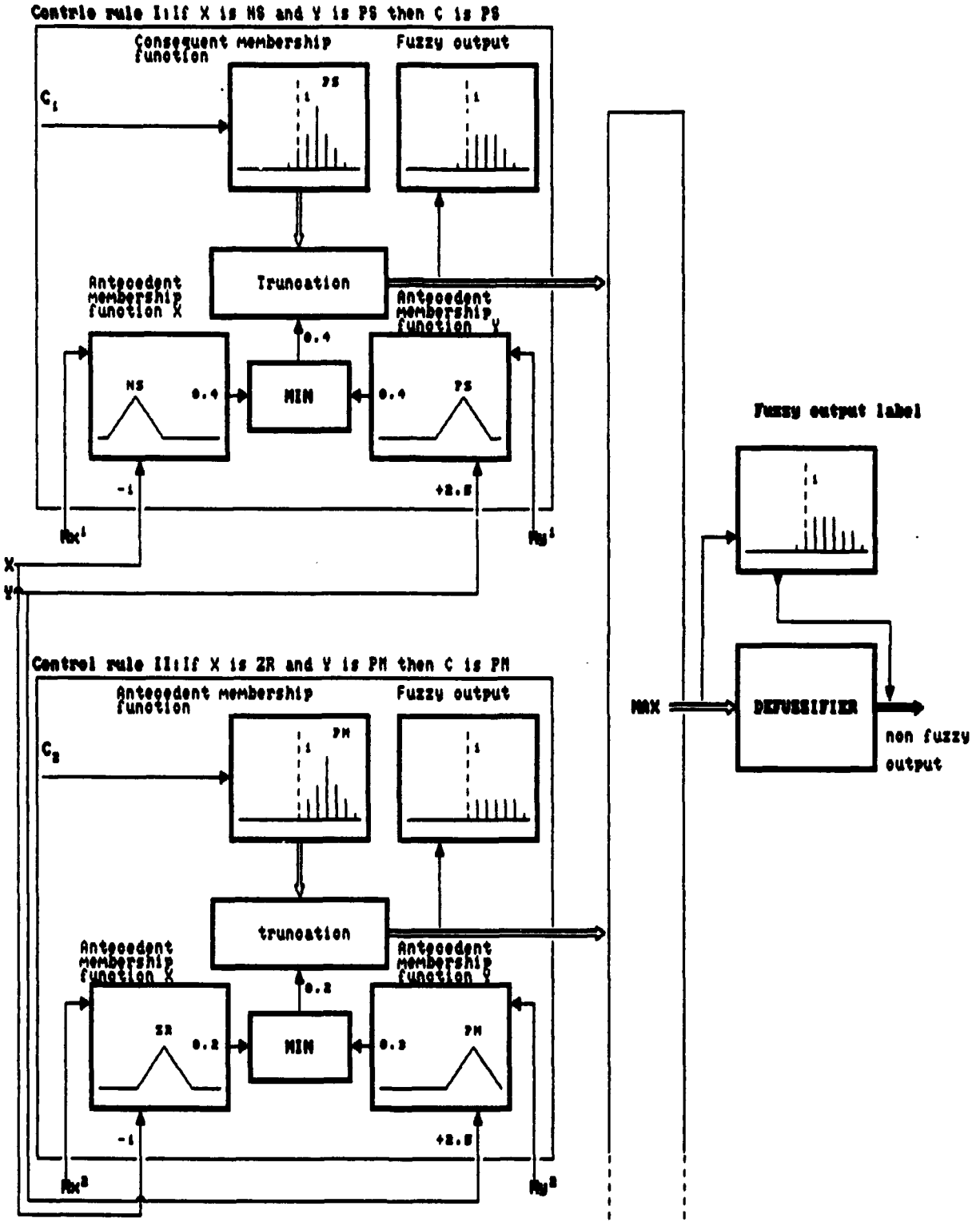


FIGURE 7.-Block diagram of two rules

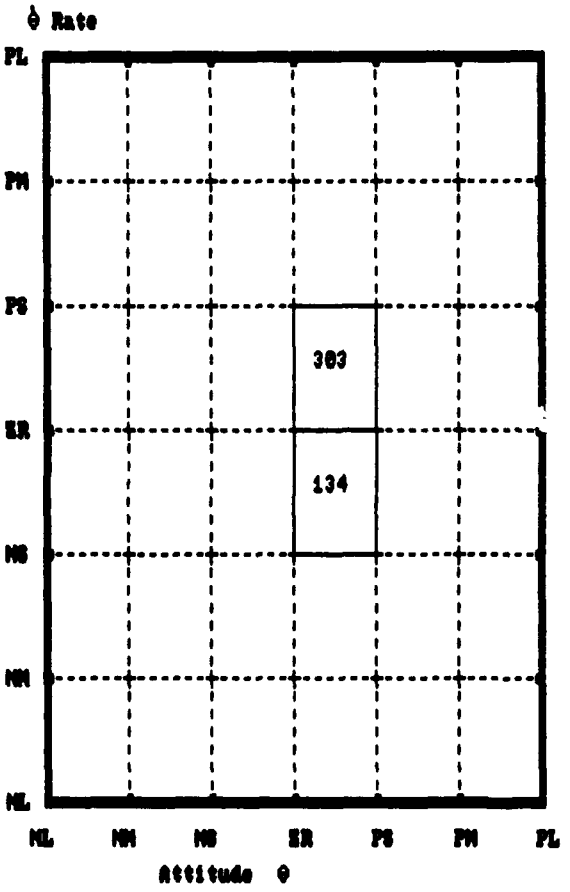


FIGURE 8.-Frequency of the cells for initial attitude 90 degrees

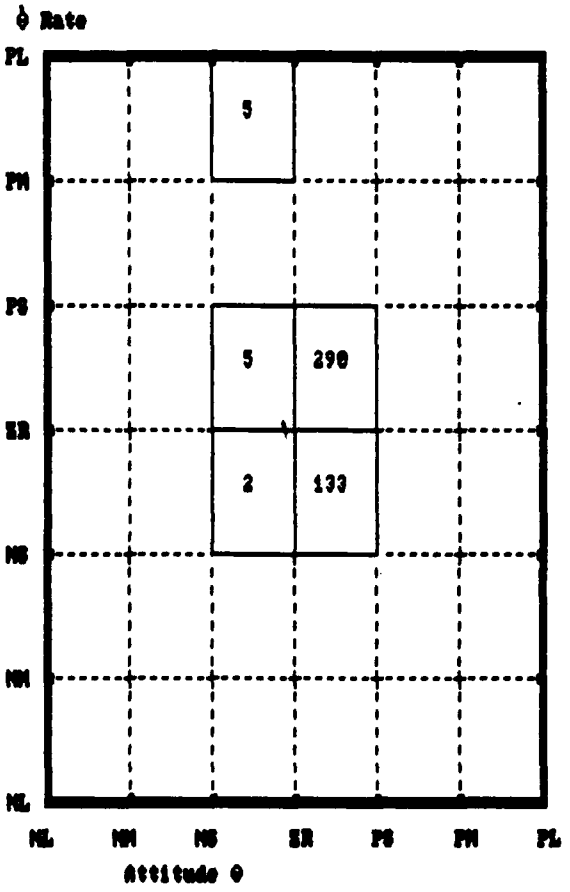


FIGURE 9.-Frequency of the cells for initial attitude 89 degrees

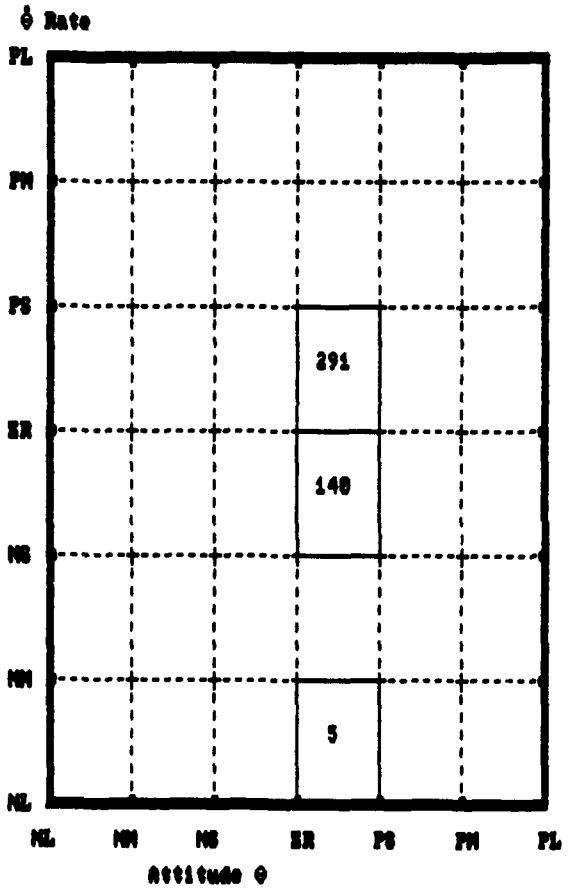


FIGURE 10.-Frequency of the cells for initial attitude 91 degrees

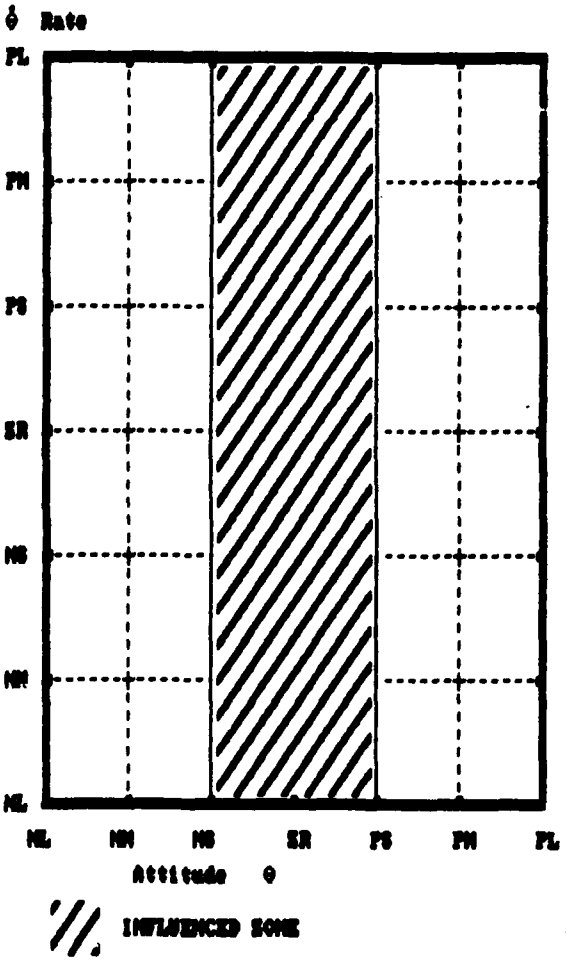


FIGURE 11.-Influenced zone

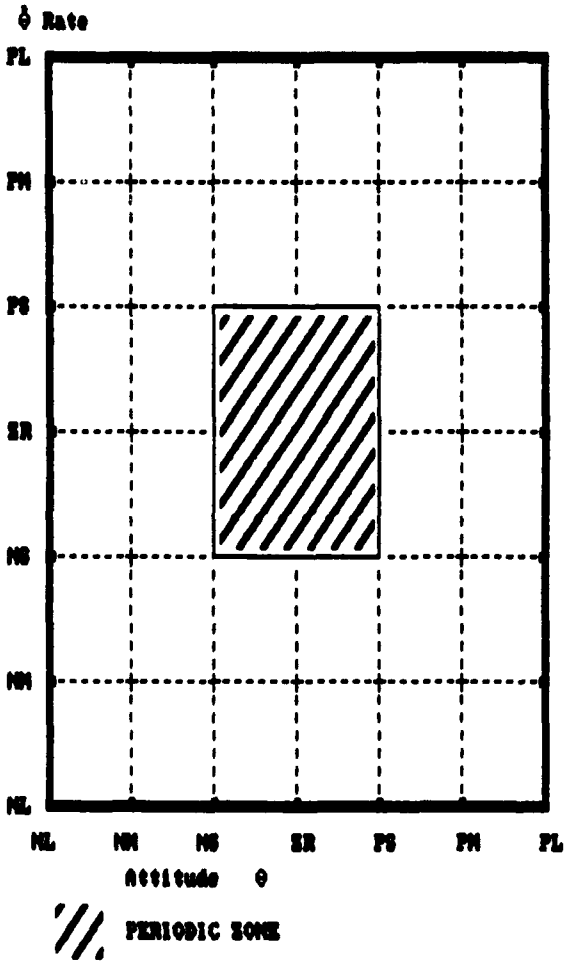


FIGURE 12.- Periodic zone

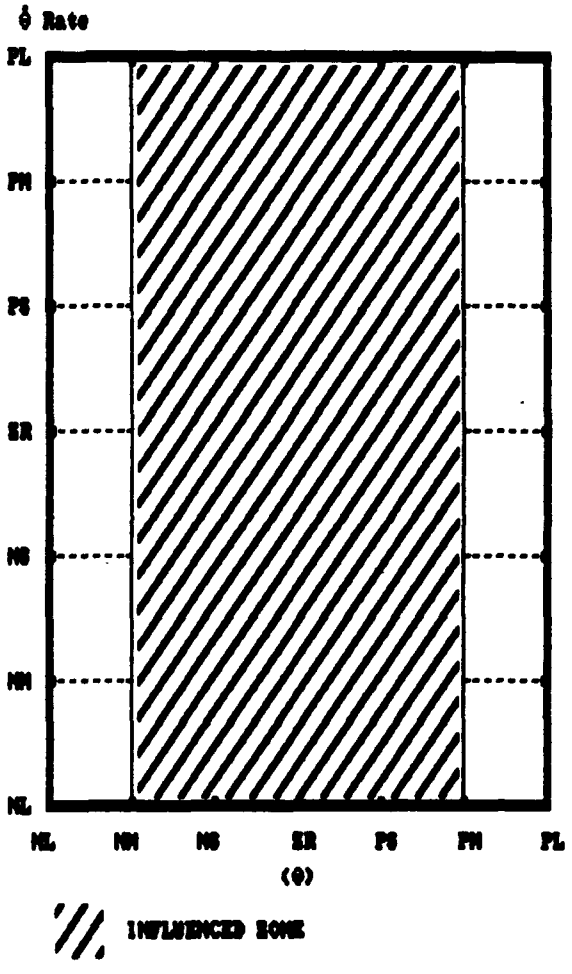
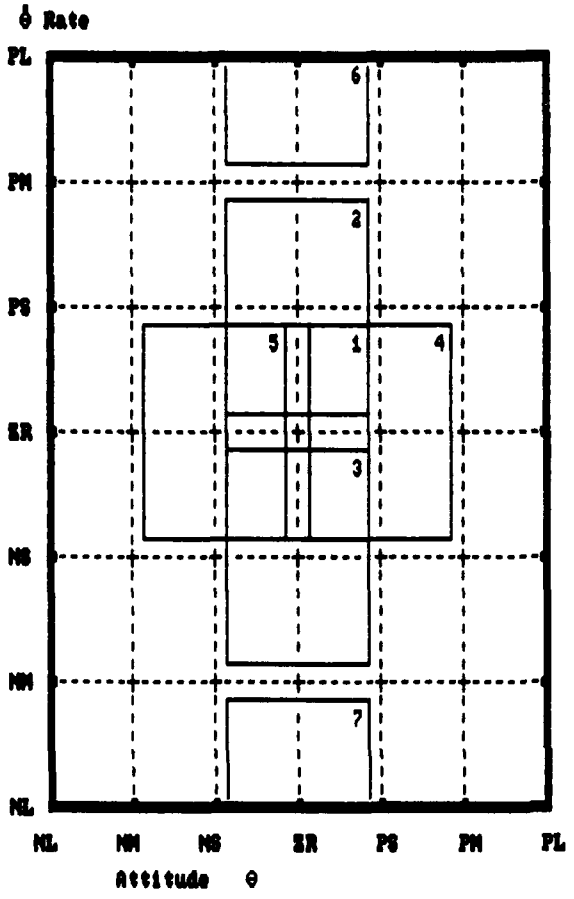
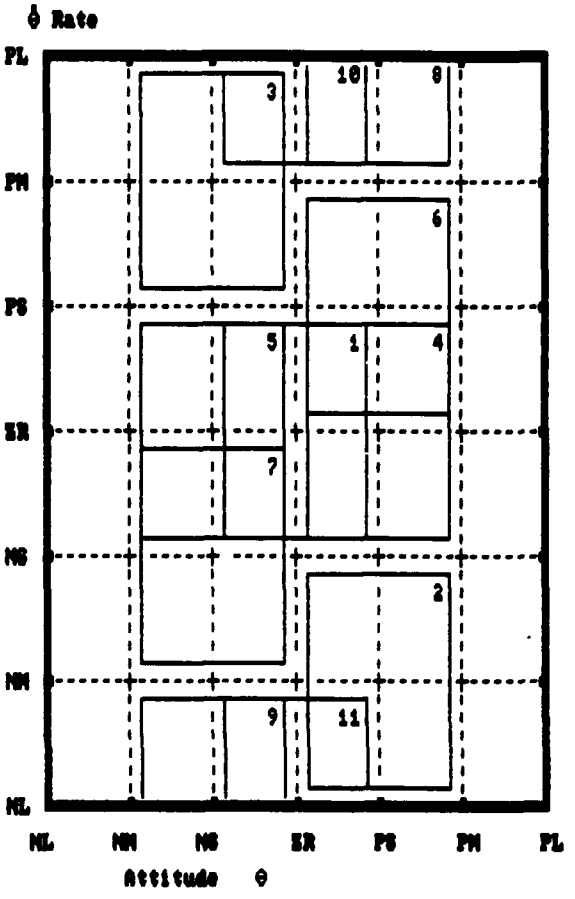


FIGURE 13.-Influenced zone



1) IF θ=ZR AND θ̇=ZR THEN δ=ZR
2) IF θ=ZR AND θ̇=PS THEN δ=NS
3) IF θ=ZR AND θ̇=NS THEN δ=PS
4) IF θ=PS AND θ̇=ZR THEN δ=NM
5) IF θ=NS AND θ̇=ZR THEN δ=PM
6) IF θ=ZR AND θ̇=PL THEN δ=NM
7) IF θ=ZR AND θ̇=NL THEN δ=PM

FIGURE 14.-Seven control rules



1) IF θ=ZR AND θ̇=ZR THEN δ=ZR
2) IF θ=PS AND θ̇=NM THEN δ=NS
3) IF θ=NS AND θ̇=PM THEN δ=PS
4) IF θ=PS AND θ̇=ZR THEN δ=NM
5) IF θ=NS AND θ̇=ZR THEN δ=PM
6) IF θ=PS AND θ̇=PS THEN δ=NL
7) IF θ=NS AND θ̇=NS THEN δ=PL
8) IF θ=PS AND θ̇=PL THEN δ=NL
9) IF θ=NS AND θ̇=NL THEN δ=PL
10) IF θ=ZR AND θ̇=PL THEN δ=NM
11) IF θ=ZR AND θ̇=NL THEN δ=PM

FIGURE 15.-Eleven control rules

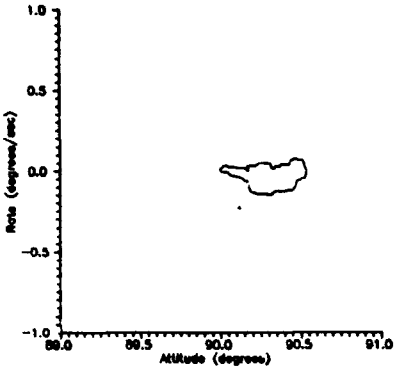


FIGURE 16.—State space rate and attitude for initial attitude 90 degrees

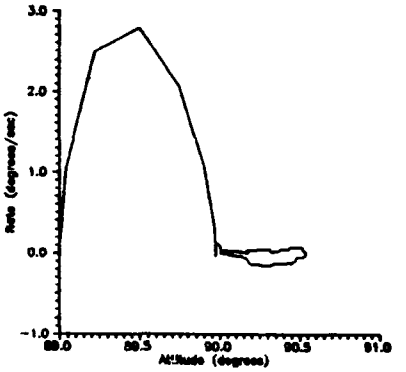


FIGURE 17.—State space rate and attitude for initial condition 89 degrees

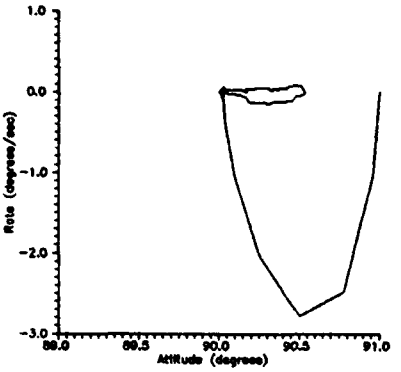


FIGURE 18.—State space rate and attitude for initial condition 91 degrees

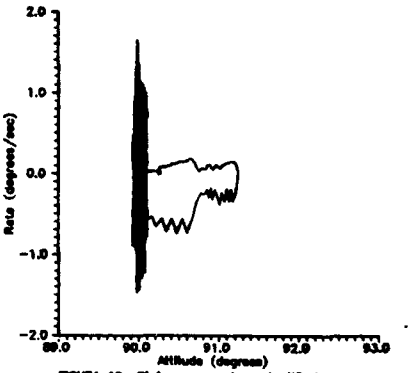


FIGURA 19.—State space rate and attitude with fuzzy controller and software simulation

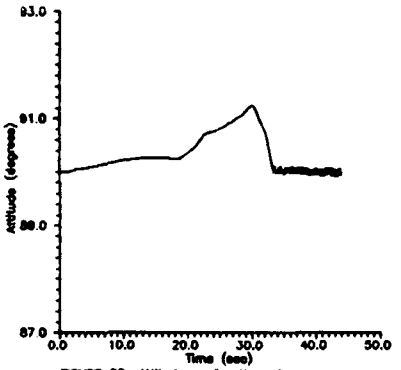


FIGURE 20.—Attitude as function of time with fuzzy controller and software simulation

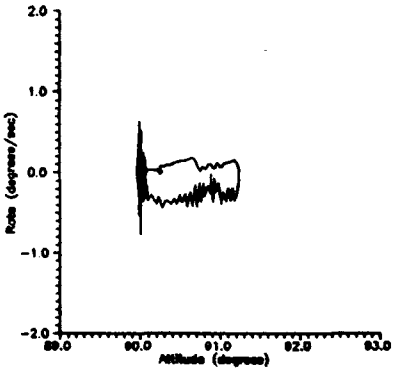


FIGURE 21.—State space rate and attitude with 13 fuzzy rules and software simulation

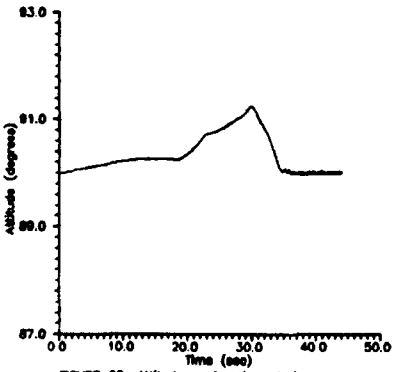


FIGURE 22.—Attitude as function of time with 13 fuzzy rules and software simulation

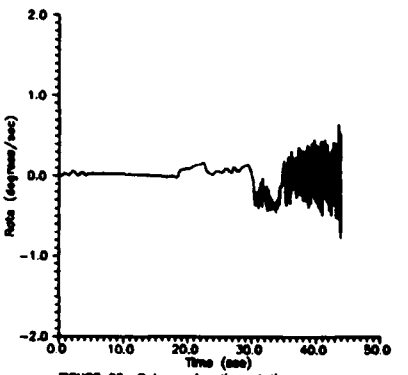


FIGURE 23.—Rate as function of time with 13 fuzzy rules and software simulation

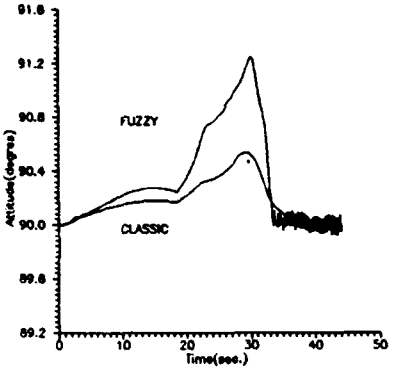


FIGURE 24

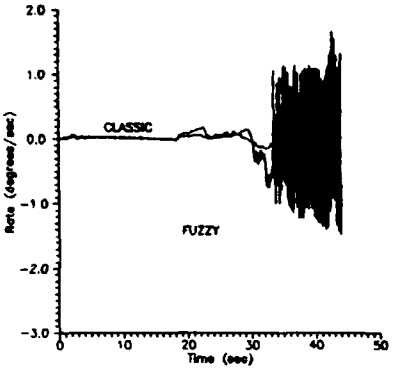


FIGURE 25

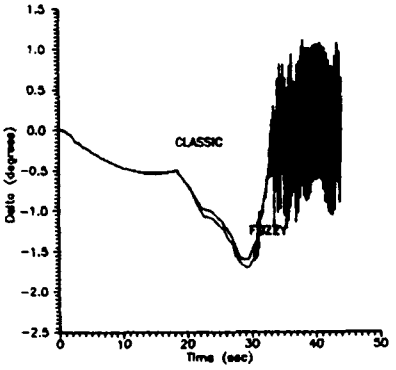


FIGURE 26

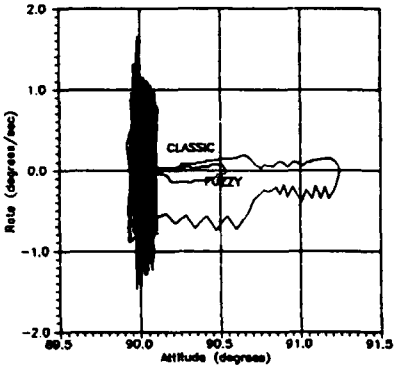


FIGURE 27

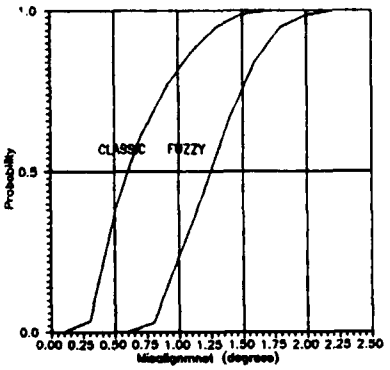


FIGURE 28.- DISTRIBUTION FUNCTION
N=200 RUNS

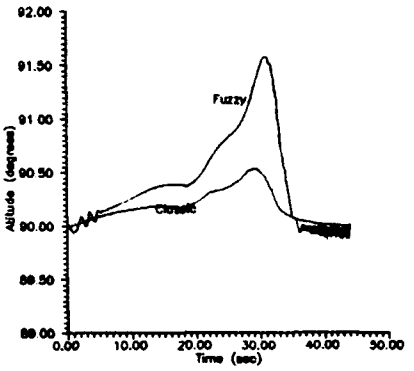


FIGURE 29.- HARDWARE MODEL

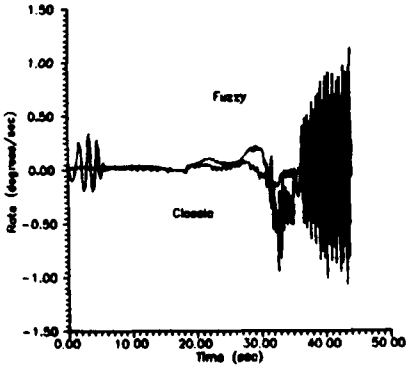


FIGURE 30.- HARDWARE MODEL

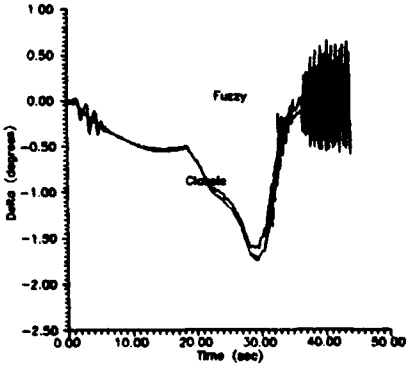


FIGURE 31.- HARDWARE MODEL

Parallel Knowledge Based Systems Architectures for In-Flight Mission Management

M. R. Bowyer
S. A. Cross
Flight Systems Department
Defence Research Agency
Farnborough
Hampshire GU14 6TD
England

1. SUMMARY

This paper focuses on the architectural approach, both in hardware and software, taken for the parallel *DMuse* project, and its application to in-flight Mission Management. This includes a description of the *Muse* Real Time Intelligent Knowledge Based Systems (IKBS) toolkit as it now stands and the alterations necessary to provide this system on a multi-processor platform to produce the *Distributed* Muse (DMuse) toolkit.

2. INTRODUCTION

The IKBS section of what was then the Royal Aircraft Establishment, Farnborough - now the Defence Research Agency - was formed to reduce aircrew workload in both fixed and rotary winged aircraft using Artificial Intelligence technology. It was recognised some years ago that a real time IKBS toolkit would be necessary to tackle the large Mission Management (MM) problems we were approaching. After some feasibility and study work had been completed a specification was given to Cambridge Consultants Ltd. (CCL) of Cambridge, England, for development. This project has taken approximately sixteen man years work to reach its current state. However, despite the growing speed of both Complex Instruction Set Computer (CISC) and Reduced Instruction Set Computer (RISC) micro-processors, execution speed was anticipated to become unacceptable for the larger knowledge bases necessary in full scale MM applications. Previous such applications have been split over multiple machines on an Ethernet network, working on separate but communicating knowledge bases, and the bottle-neck was found to be both execution speed and communication bandwidth. The solution to this is seen to be a multi-processor hardware platform, which we believe can alleviate both these problems.

3. MUSE

Muse is a powerful real-time Artificial Intelligence toolkit produced by CCL to a specification developed by DRA. It consists of a full development environment, including a structured editor, a run-time browser and a debugging facility. Various language systems are also provided to give greater efficiency with differing types of application. Muse is ideally suited to applications such as fault diagnosis, intelligent scheduling, condition monitoring and knowledge based simulation. It has also been designed with embedded systems use in mind.

The Muse package contains an integrated set of knowledge representation language modules. These comprise of a flexible frame language, a forward production system (FPS) and a backward chaining system (BCS) which are supported by and partly written in *PopTalk* - an object oriented procedural language. This is an implementation of the POP language, written in C, but which includes the object oriented programming tools used in the *Smalltalk* language. The frame system sitting on top of this includes features which allow applications to be easily written for real-time execution, such as procedural attachment and message passing.

In the FPS and BCS rule-based reasoning systems, rules can match and modify objects in databases. Those in the FPS fire in a data-driven manner using the efficient RETE matching algorithm, while rules in the BCS are goal-driven and provide an automatic backtracking search. BCS rule systems are best known for their use in *Prolog*. Both the FPS and BCS use the same database structure and can therefore easily share information.

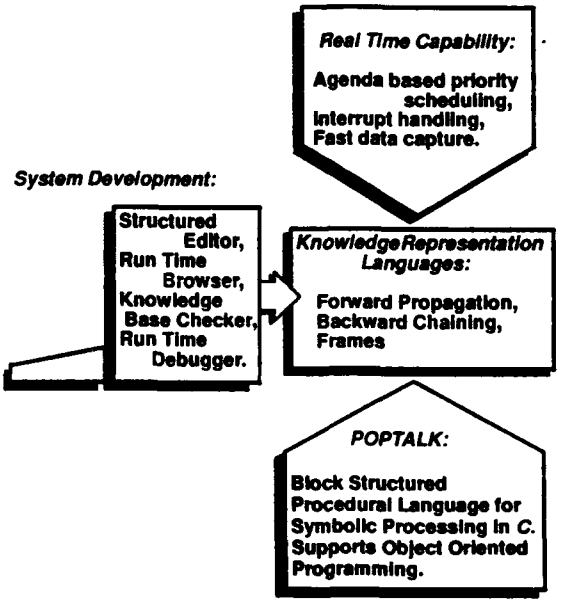


Figure 1: The Muse Toolkit

Muse also provides support for the modular development of applications. These facilities include databases, knowledge sources, notice boards, agenda-based scheduling, collections and data capture channels. Also very important in real-time applications is the design of the knowledge sources, as it allows the safe interruption of an existing activity to deal with a new event.

The development tools provided with the Muse package include a windows-based structured editor, a run-time structure browser, and the MUDDLE run-time debugging system. The structured editor provides a creation and manipulation tool for the object definitions which make up all Muse applications. Even rules are treated as objects. Details of the actual code can be 'folded' away out of sight where necessary to give the user a view over the entire structure. An interface to the EMACS editor is also provided, as is an interface to the UNIX Source Code Control System (SCCS), to maintain a full backtrace of application

development. The browser tool provides the same user interface as the editor but allows the examination of run-time data structures. This is also a convenient display screen from within applications. The MUDDLE debugging system provides all the usual tools, such as stopping execution at marked positions, invocations or invariant violations; state examination; tracing execution and single stepping. A static knowledge base checker is also provided to spot inconsistencies between an object's declaration and applications before compilation.

4. THE NEED FOR ENHANCEMENT

Although initial, smaller systems built with Muse on Sun 3 machines had proven fast enough, as more complex systems were developed by ourselves and other Muse users it became apparent that single machine performance would not be adequate for full scale real time embedded applications. At first we hoped to solve this by splitting systems over multiple machines on a network, or by improving hardware performance (i.e. moving to SPARC), but this just gave a reprieve from the problem. We realised that for full MM application solutions, multiple high-performance processors, preferably closely coupled, would be the best solution. Thus, the first decision to take was how to split the Muse environment over multiple processing areas.

Low level parallelism was considered briefly. The finer grain option was to implement a parallel version of the RETE match algorithm (an efficient implementation of the matching task for a production rule system) after [1]. At a slightly higher level, the development of a true concurrent object-oriented language with a single object space and implicit load balancing and scheduling was the second option. Work of this nature includes ABCL/1 [2], Concurrent Smalltalk [3] and REKURSIV [4].

Both of these approaches are high risk, requiring major rewrites to the core of Muse. In addition, the concurrent objects system is probably unsuited to the development of MM systems. This is because it removes the notion of processor and process and could encourage inefficient design when implemented on the limited inter-process bandwidth systems likely for MM. However, the lower risk design proposed below allows the first option to remain as a possibility for subsequent development.

This left higher level parallelism as the solution of choice, given earlier success with speeding up Muse applications. Muse would need to provide support for the design of efficient parallel implementations of MM tasks, so the various design metaphors available and their applicability were examined.

5. METAPHORS

In the real world, groups of people work on large and/or complex tasks. There exist various ways of organising their work, and these real world systems have provided "metaphors" for organising distributed AI. Lesser & Corkill [5] classify these metaphors along two dimensions. The first is related to the level of data uncertainty in the problem, from completely accurate to functionally accurate (uncertain). The second indicates the level of control uncertainty, i.e. what individual processes should be doing and what results they share, from nearly autonomous to cooperative. Thus there are Functionally Accurate, Cooperative (FA/C), Completely Accurate, Nearly Autonomous (CA/NA), Functionally Accurate, Nearly Autonomous (FA/NA) and Completely Accurate, Cooperative (CA/C) systems. However, Lesser and Corkill point out that most systems would be classified as either FA/C or CA/NA because the presence of control or data uncertainty tends to exacerbate the other.

Under these headings, there are a number of specific metaphors. The Contract Net Protocol [6] and the Ops Room metaphor [7] fall squarely under CA/NA while Multi Stage Negotiation [8] uses the Contract Net Protocol for initial negotiation but the subsequent rounds are claimed to give an overall flavour of the FA/C approach.

The following sections describe each metaphor in more detail and examine their relevance to the MM task.

5.1 Functionally Accurate, Cooperating Systems

Lesser and Corkill produced the above classifications of distributed systems during their research to apply the potential benefits of distributed computing more widely, i.e. to systems that were not easily translatable to CA/NA (such as distributed databases and process control). Tasks such as sensor fusion, air traffic control, power network control and those involving mobile robots require nodes to work on possibly incomplete, incorrect and/or inconsistent data. These tasks are also characterised by their expressability as search processes which require multiple local partial decisions to guide searches, which have a number of paths to adequate solutions and which operate at multiple levels of abstraction.

It is their contention that having multiple nodes work on different parts of a problem and share their local hypotheses with each other exploits the latter set of characteristics to overcome the former. Nodes must therefore be able to detect inconsistencies in other nodes' hypotheses, integrate consistent hypotheses and use these results to refine their partial solutions. Experimentation has indicated that nodes should share higher rather than lower level hypotheses and that it is useful to share control information about a node's state to avoid duplication of effort.

Since Knowledge Based Systems are usually associated with problem solving in the presence of inconsistent, incorrect and/or incomplete data, Durfee, Lesser and Corkill [9] have explored the use of the knowledge based Hearsay II architecture [10], developed for speech recognition, within nodes. This has the necessary structure to work at differ.:t levels of abstraction in a data directed hypothesize and test search. In applying a three node version of the original Hearsay II to contiguous fragments of speech a 60% speedup was achieved over a single node version. Although the same results could be achieved more efficiently with a 60% faster processor, a more significant result was that performance degraded gracefully with up to 50% of inter-node messages (i.e. shared hypotheses) being lost. This supports FA/C's claimed abil:ty to overcome the problems of incomplete data.

The Hearsay II architecture (with different knowledge from the speech recognition version) was used in a multi-node architecture forming the Distributed Vehicle Monitoring Testbed (DVMT). A number of vehicles move over a geographical area. There are a number of sensors over the area and each is monitored by a computing node. Each node can, therefore, monitor partial vehicle tracks. The intention is to provide a picture of vehicle traffic. This is achieved by having nodes share hypotheses and local goals. The testbed allows variation of sensor overlap and level of hypothesis exchanged. In this case the levels are raw sensor data, groups of related signals, collections of groups related to one vehicle type and patterns of groups to identify vehicle formations. It was work with the DVMT that indicated high level hypotheses and local goals (states) should be exchanged to improve performance [9].

The speech recognition and DVMT experiments could be classified as one- and two-dimensional sensor fusion. They have had sufficient success to suggest that FA/C would be a productive approach for the three-dimensional sensor fusion required for the Situation Assessment task of MM.

5.2 The Contract Net

This is a communication protocol with application to dynamic task allocation. A manager node broadcasts a task announcement to contractor nodes. This contains a node eligibility specification, an abstract description of the task and required bid formats. This broadcast may be limited to a subset of nodes or even a single node. There is a fixed time window in which nodes should bid for the work. The received bids are assessed by the manager and a contract awarded. Bid processing is application specific, using strategies such as accept the bid requiring least information, or the first bid to arrive. Similarly, contract nodes have an application specific task acceptance strategy. The winning contractor then performs the task. Information messages are used for general communication between manager and contractor and results are available through interim and final reports. The contractor may also subcontract out portions of the task.

The key to use of this metaphor is devising a common inter-node language for bids which allows efficient exchange of information for task descriptions. This reduces the potential problem of message overhead. The presence of bid deadlines means that it is suited to real time operations where desired system response times are known, provided that the message overhead problem can be solved.

Smith [5] gives the example of a distributed sensing system which consists of a collection of sensor and processor nodes spread over a relatively large geographic area. It attempts to construct and maintain a map of vehicle traffic in the area.

In this application, the processor nodes are assumed to have extensive signal processing capabilities but no sensors, while the reverse is true for the sensor nodes. Therefore, one situation requires the processor node to announce tasks to receive sensor data. The eligibility will require that a node is located within some geographical area and has certain types of sensor. The bid specification will require the sensor node's exact location as an entry. The task abstraction gives the type of task (in this example it is called "signal") and the location of the manager (processor) node. (This latter information could be used by the sensor node in deciding which of a number of signal task announcements to accept.) The manager's evaluation of the resulting bids could be based on the proximity of a sensor node to an ideal location.

This example is similar to the DVMT of FA/C above, so the contract net might be considered for use in Situation Assessment. However, the data uncertainty associated with SA could be problematic for a CA/NA metaphor. Nevertheless, the explicit real time considerations embodied in the bid deadlines are an important feature for on-board systems if the message overhead can be reduced sufficiently.

5.3 Multistage Negotiation

This metaphor is used for planning in a distributed environment where the problem may be over-constrained i.e. not all constraints are satisfiable. It enables nodes with a limited communication bandwidth and no centralised control to acquire knowledge about the effects of a local activity on non-local state and to modify their behaviour accordingly. Nodes initially perform local planning and undergo a single phase of Contract Net negotiation where they announce tasks related to furthering their local plans. After this phase, each node has a set of ordered plans to achieve its local partial goals. Subsequent rounds of communication are to confirm that its choices do not conflict with those of other nodes. Conflicts cause a node to reevaluate its set of plans and to seek confirmation of these new alternatives. Termination of the negotiation can be globally implemented by token passing, or over constrained problems can be terminated in a diffuse manner with local nodes performing an "irrevocable" commitment of resources, usually if a node finds itself retracting and reinstating a plan with no new data appearing.

The example application is maintaining point-to-point computer communication circuits in a simplified wide area "backbone" network. Each node is responsible for a geographic area which contains a number of computer sites and links between them. Each link can support a number of circuits. A problem occurs when links involved in circuits fail. Nodes controlling the area where a disrupted circuit terminates perform initial planning to identify alternative link routing for that circuit. This uses the Contract Net negotiation phase where task announcements are put out to adjacent nodes asking for links to re-establish the circuit. (Note that nodes are not aware of link patterns in other areas, nor of all circuits that are disrupted.) Subsequent rounds of negotiation verify that the solutions generated by the initial planning do not overload any links.

Tactical Planning is a major task in MM. In a similar fashion to the preceding example, there are limited resources available for plans e.g. time. However, the plans of nodes in a distributed tactical planner will probably interact with every other node rather than those geographically adjacent, so negotiation may be more complex than for the computer circuits example. Nevertheless, Multistage Negotiation could be a useful metaphor for Tactical Planning.

5.4 The Ops Room Metaphor

This is based on the air defence ops room and is like the Contract Net but with a fixed task allocation hierarchy. Nodes are responsible for objects on a centralised tote board. They report to nodes above them in the hierarchy and allocate tasks to nodes below. The tote board is a potential bottle-neck but the original problem required nodes to maintain relatively few objects on the board, each of which had a high semantic content. In the language of the Contract Net, there was an efficient common inter-node language.

This metaphor has obvious application to ground based MM, so it is worth bearing it in mind when deciding on requirements for parallel Muse.

In summary, both FA/C (including Multistage Negotiation) and CA/NA metaphors (Contract Net and Ops Room) are potentially suited to the MM problem. Muse must provide the design architectures to support these metaphors. The facilities proposed in *Language Enhancements* below do not do so directly (in terms of providing task announcement objects etc.) but the necessary machinery will be available to use these metaphors with a small amount of additional design work e.g. specifying task announcement objects. After describing the possible methods for extending Muse towards these goals, section 7 describes this machinery and the section 8 demonstrates how each metaphor could be implemented within this framework.

6. POSSIBLE METHODS OF ENHANCEMENT

6.1 Concurrent Objects

These systems provide a uniform object space in which each object of the application possesses a unique global reference. Object access is equivalent irrespective of where the request is made, and scheduling and load balancing are handled as an implicit function of the language. Unfortunately, the uniformity this brings to an application might encourage programming styles unsuited to the processor architectures that are likely to be used in airborne MM systems, by exhibiting a single memory space to the developer, rather than reflecting the limitations of inter-process bandwidth. It would also involve an extensive re-write to Muse, as the stack based model of PopTalk is not ideal as a basis for a concurrent language, and the performance penalties of concurrent languages should also be borne in mind.

6.2 Distributed Rule Systems

Such a system would split up knowledge bases over processes. To extend the functionality of both the Forward Production System and the Backward Chaining System, it would be necessary to support variable bindings across processes, and possibly processors. It would be necessary for the BCS to support backtracking across process boundaries. This would in turn necessitate a high degree of coupling between processes, due to the control implicit in the rule systems, and would probably be better implemented as a superset of the extensions described above, involving an extensive Muse rewrite. Also, the metaphors chosen for MM applications do not map well onto this architecture, and it is difficult to describe its intended behaviour, either diagrammatically or verbally.

6.3 Shared Inter-process Databases

The ability to share databases, or blackboards, amongst knowledge sources already exists within a single Muse process, using either the *interests* mechanism or by objects maintaining references to individual databases. It would therefore be possible to extend this so that each Muse process could also access databases shared with other processes, while still keeping local databases invisible to the others. This would allow the developer to view the system either as data sharing or as message passing, by considering the database as a message slot into the process, from which data can be disseminated by using demons and production rules.

Conceptually, inter-process databases can be viewed as a single database external to the Muse processes that use it (see Figure 2.)

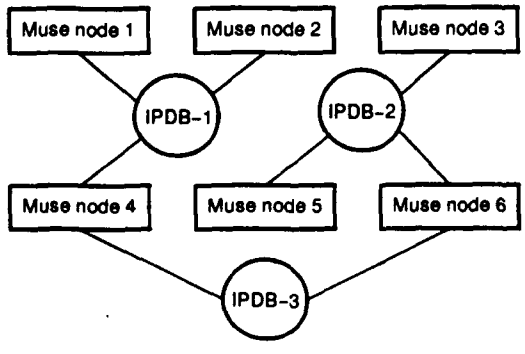


Figure 2: Conceptual Implementation

In practice, however, each process would have a local instantiation of the inter-process database, with object assertions being broadcast to each other in the group (see Figure 3.)

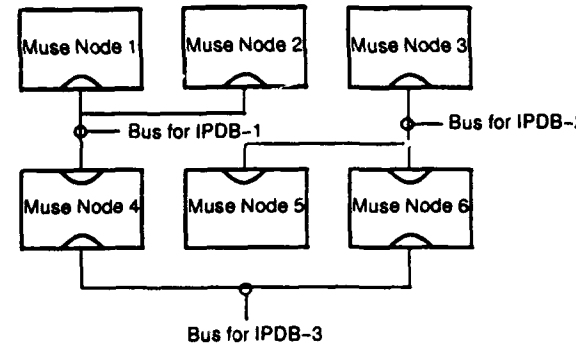


Figure 3: Practical Implementation

Visibility could be controlled by defining access rights of processes to databases, such as read/write, read only, etc. Each process would hold its own definitions of the *schemas* (objects) of all instances the application intends to assert into, or read from, an inter-process database, so that a minimal amount of data would need to be broadcast to reconstruct an object asserted by another object. It would also be necessary to allow only the creating process to modify or retract an object, to remove the need for handshaking and locking functions. This would prevent a process changing another's view of an object, although of course a process could make its own copy for modification.

This proves to be a conceptually simple extension to Muse, allows the best support of the Metaphors suggested above and requires considerably less work to implement than the alternative approaches that were investigated. Therefore, it has been chosen as the architecture for DMuse.

7. LANGUAGE ENHANCEMENTS

The use of the current Muse language over a distributed system, supporting the Metaphors described above, will require the addition of some extra facilities to the developer. The following enhancements have been put forward to this end.

7.1 Object Broadcasting

It is expected that there will be two parts to the object broadcast facilities essential to the development of DMuse. The first would monitor assertions, modifications and retractions of locally created objects on the inter-process database and broadcast them to the other interested processes. It should be possible simply to extend the current database access methods to perform the required actions. The second part would monitor such incoming messages from other processes. It would first have to decide whether the broadcast effects a database it is interested in, and, if so, make the appropriate changes.

This also supports the dynamic creation of inter-process databases, automatically endowing it with the necessary methods for broadcasting events to other processes and declaring the database to the system. However, a mechanism would need to be provided to transfer efficiently an inter-process database's contents when a process expresses interest in it.

7.2 Database Demons

Within the current Muse, demons are designed only to monitor the creation and slot updates of objects, whether in or out of a database, and forward production rules must be used to monitor the addition, retraction and modification of objects in databases.

It has therefore been suggested that a "database demon" be defined, providing a more efficient method of monitoring changes to an inter-process database.

7.3 Removing Inter-process Databases

If we are going to create inter-process databases, we also need to remove them. This actually requires three actions. The process has to retract all objects it asserted onto the database, and broadcast these retractions. It must remove the database from its list of monitored inter-process databases, so that no further events for it are received. Finally, it must notify all the rule sets and database demons that are monitoring events on the database. This means that the demons should be freed for the garbage collector and the database removed from the rule set's interest slot.

7.4 Sections

For us to be confident about splitting an application's knowledge sources and allocating them to separate processors, it would be very useful for the knowledge sources and their associated support functions to have been compiled into *sections*. Sections provide a tool for controlling the visibility of declarations, and are already supported in Muse, but without any support for compiling knowledge sources within them. By allowing this, knowledge sources are less likely to be dependent on code defined elsewhere in the system, and transferring them between processors becomes easier.

7.5 Application Loading

Loading and running a distributed application over multiple processors, consisting of many separate processes - some written in DMuse, some in C or some other conventional language - can be a non-trivial problem. The toolkit therefore requires a loader utility that takes the application and an allocation description and performs the necessary loading tasks, including the configuration of communications software for physical links and finally starting the system. This is a far more significant problem than the previous single processor target machine loader.

7.6 Fault Recovery

Since DMuse is specified for use in real-time embedded systems for avionic MM, some form of software fault recovery - other than hardware multiple redundancy - should be included. Should a processor fail, it is not plausible to restart the entire system, and re-allocate processes. However, the proposed architecture does offer a means of recovering all of the external data that was visible to the now dead process. It should therefore be possible to restart this process on a spare processor, or the least loaded processor in use and, using its list of inter-process databases, request the necessary data to be broadcast to it. Providing this initial list is the same as that in use as the point the original process died, this should allow the system to degrade gracefully rather than being destroyed. Given such an addition to the architecture, a process which is critical to the welfare of the entire application need only deposit internal data regularly with a process on a separate processor to be sure of its ability to continue after a failure. Such a depository might be a dedicated, but otherwise passive, C process that behaves like an inter-process database, making the entire system more resilient.

8. METAPHORS IN DMUSE

This section describes how the various metaphors for MM could be implemented using the proposed software enhancements of DMuse.

8.1 FA/C

Nodes that can communicate with each other have shared databases. Hypotheses are broadcast simply by inserting them in this database. At the local recipient level, nodes can monitor the insertion of hypotheses by using demons or production rules.

8.2 CA/NA

For this class of metaphor, nodes usually transmit data and wait for a response. Again, a shared database would have the message inserted in it and the recipient would use demons or rules to detect it. However, the original sender would have to explicitly suspend its activity rather than implicitly waiting for the response to appear in the database.

8.2.1 Contract Net

This requires a database shared by all nodes for contractor-wide broadcasts. The initial task announcement is inserted into the database by the manager and the contractors then reply by inserting their bids in the same database. Upon award of a contract, the manager and contractor can set up their own shared database for subsequent direct communication.

An advantage of this implementation is that contractors can see whether or not they won the contract as soon as it is awarded. A less useful divergence from the protocol is that managers cannot broadcast to a subset of contractors unless private databases already exist, although since the number of DMuse processes is expected to be static, a filter field could be added to the task announcement to overcome this.

8.2.2 Ops Room

This is similar to the contract net in that there is a database shared by all nodes - the tote board. It differs because the nodes are organised in a hierarchy with one database shared between a parent and all its children. The static number of DMuse processes is not a problem here since the hierarchy is also static for all run-time.

8.3 Multistage Negotiation

This uses the Contract Net metaphor for the initial phase, which is supported as described above. However, the lack of contractor-subset broadcasting is probably not a problem here since subsequent rounds of negotiation with specific sets of nodes may be required. The sets will be known at design time e.g. nodes communicate with others in geographically adjacent areas in the computer circuit example. Thus the required extra databases can be specified in the design.

9. DEVELOPMENT TOOL ENHANCEMENTS

No matter how good the Muse toolkit itself might be, without a sophisticated development interface it is unlikely to be used to its full extent. The current tools, described above, have proven very useful for developing single threaded applications, but the shere size and complexity of MM applications are likely to entail multiple developers as well as processes and processors. It should also be noted that debugging, in a non-deterministic distributed environment, of multiple processes and processors, is considerably more complicated than debugging a single process. The following enhancements have been proposed.

9.1 *The Structured Editor*

Support for multiple development streams, requiring the integration of the work of separate individuals as they complete their parts of the application, suggests that the current Muse method of anonymous "*.m" files is inadequate. Instead the DMuse structured editor should be supplied with a set of module

pointers (file names) to be built into the application when the editor is invoked. Each developer can then modify their own code as with the current structured editor, and when the session is terminated, their source is written out in a format that can be referenced by other developers. The UNIX *sccs* source code control mechanism could also be used at this level to provide a development backtrace and logging mechanism.

9.2 The Browser

The current Browser works with the same structured folding editor technique, but displays objects from within a Muse application. For DMuse, many of the objects assigned to inter-process databases will have been created in other processes, and so viewing would have to cease where the value referenced into another process. The DMuse browser should therefore have the ability to follow these references.

These values are based on a snapshot of the object's value at the moment the display was initiated. Displaying complex data structures can be computationally expensive, since it has to create a proportionately sized data file. To achieve greater efficiency, alternative browsing methods should be supported. These would include lazy evaluation - only updating the value of a slot when its fold is opened, and re-evaluating only when demanded (possibly by closing and opening the fold again) - and dynamic updating - re-displaying the value whenever it changes.

The DMuse browser should also have the capability of displaying objects from different processes and processors in multiple windows, as well as allowing folds in one window to be opened into another.

9.3 The Debugger

The current system only allows *trips* (break points) to be set on rules. Within DMuse this should be extended to all runnable objects, and therefore also requires the browser to be able to display these objects, including procedures and methods. Trips should also be able to halt one or more processes as soon as is practicable, allowing the developer to examine a frozen system.

Single stepping also becomes a problem in DMuse, as it needs to follow execution across process boundaries so that the developer can watch the effects of an event across a distributed application. It should also be possible to single step processes affected by a single event independently of each other. The ability to indicate the current tripped point in a source display should also be included.

10. HARDWARE EVALUATION

It was decided at an early stage of this research that we would concentrate on Reduced Instruction Set Computer (RISC) architecture machines. This was due to the increased performance they provide and the even greater increases in performance that they promise over Complex Instruction Set Computers (CISC), such as the Intel 80486 and Motorola 68040. The sort of features generally found in RISC processors are:

- All machine instructions are executed in a single processor cycle - there is no micro-code.
- The processor recognises far fewer and simpler instructions than a CISC, hence the name. This doesn't prove too great a problem, as research shows that 80% of software uses only these simple instructions and the speedup produced by not having to decode micro-code easily makes up for the extra time spent running multiple instructions for a more complex task.
- Operations are only carried out on registers, with memory only accessed for loads and stores.
- Instructions come in a fixed length format. This simplifies the fetch/execute cycle and the supporting hardware.
- The processor has a large register set and/or 'register windows' to improve compiler efficiency. This allows procedure calls to have their own set of registers, usually with some overlapping with the previous, calling procedure to pass data between them, a local set, and some shared with called procedures.
- Memory fetch pipelines and/or cache memory are employed so that fast, single cycle processors can be used with slow main memory.

RISC processors are also generally more reliant on software support tools than CISC for several reasons:

- The reduced functionality of the RISC instruction set means that coding in assembler is far more difficult and compilers are more necessary.
- Many functions that are removed from the instruction set are still used commonly enough that they must be replaced with optimised library functions. RISC processors are often supplied with libraries for functions for integer multiply and divide or array access, for instance.
- Some features of the RISC processor can only provide performance enhancements if they are utilised correctly by a compiler. Examples of this are register windows, which must be mapped onto procedure calls and delayed branches, which can improve pipeline performance. Superscalar architectures, which allow each separate section of the processor to operate in parallel, require instructions to be fed to it in groups of one for each part to achieve optimum performance.

Most RISC compilers also contain optimisation techniques designed specifically for their target processor, a process which is harder to develop, and so slower to appear, on CISC platforms.

The evaluation of both the above software enhancements and of the possible target machine architectures was undertaken as a study contract ending in April 1990. Although the decisions reached about any changes to the Muse system would remain esentially stable, it was noted when the contract started that the microprocessor market place was so volatile that any conclusions might immediately be invalidated. For this reason it was decided to compare each processor on how its architecture supported the Muse virtual machine model, and the level of performance expected for Muse on each processor running at equal clock speeds. This last part required some normalisation of the benchmarked figures which, since they took into account only clock speed and not memory speed, bus bandwidth or co-processor performance, was not particularly ideal.

However, the Heines and Jalics benchmarks [11] described below and used here were implemented with loops of small sections of sequential code which, if able to fit into the processor's cache, gave it a great advantage as it had no need to make external memory references. Once the cache size is exceeded the performance loss becomes very evident as the external memory reads take up more time. Muse is a very large program that seldom operates sequentially when it is interpreting Poptalk code, and so where some processors might have this advantage with the benchmarking, it probably would not be reflected with Muse itself. Since the Muse program used for the

benchmark was a data fusion example, which read new data on each iteration, a data cache was ineffective. Also affecting this is the size of the instruction pipeline, since it must be flushed at a context switch - the longer the pipeline, the longer it takes. Despite all this, we do believe we were able to estimate with some accuracy the expected performance of Muse on each processor.

The Heines and Jalics Benchmarks were chosen for their ease of porting and highly detailed output on the performance of each different type of C instruction. Rather than relying on only computation-intensive tasks or 'synthetic' benchmarks that try to include a typical instruction mix from a range of applications, these tests measure cycle times over a set of more than 400 common C 'instructions'. This is achieved by measuring the time taken to execute the smallest components of a high level language program, averaged over thousands of repetitions of a single fragment of C code. The Heines and Jalics suite also includes a set of 12 more commonly used benchmarks for comparison, such as Dhrystone, Quicksort, Sieve, etc. Some care is taken to ensure that optimising compilers don't remove these pieces of code as redundant, which would result in zero run times. The advantage of all this is that the result is not a single figure, but rather a complete set of diagnostic tables that can be used to asses and compare particular compiler/processor combinations.

Using normalisation proved to be a wise decision, however, as nearly all the processors compared have since been superseded by better and faster models. The problems of finding board level products, and the necessary software for Muse to be ported to and to run on have also in most cases been alleviated since then. Even so the results given for the SPARC processor are now not particularly relevant, as the SPARC design is being manufactured by various different companies using different techniques. Current versions include a super-scalar implementation, which implies that more than one instruction will be being processed per cycle - as the i860 already does - and this would therefore not produce figures in line with the benchmarked SPARC for its respective clock speed. Equally, MIPS are now about to release the R6000, which uses a compatible instruction set but a different architectural design.

For this reason we will not enter into the results in any great detail, but will instead try to explain why the architecture chosen was considered better, for our use, than the others tested.

11. PROCESSOR ARCHITECTURES

The processors chosen for evaluation were the Sun SPARC, the MIPS R2000, The Intel i860, the Motorola 88000 and the Advanced Micro Devices Am29K. We will first discuss the specification of each, and how this relates to porting Muse to it. A description of the processor architecture is then given, followed by a review of available products for our use at the time. An overview of the position of each in the RISC processor commercial marketplace is also given as a pointer to likely future developments. They were compared to the "standard" Muse system running on a SUN-3 machine using a Motorola 68020 at 20Mhz, and all run-times were calculated as a percentage of those measured on it. The timing figures are given later.

11.1 The Intel i860

The benchmarked system used was a plug in accelerator card for an IBM PC compatible machine called the "star860" using an i860 running at 33Mhz. This was the highest clock speed tested, and so gave it a disadvantage in the normalisation process. The C cross compiler supplied was ANSI rather than System V compliant, and did not include sufficient support to port Muse to it for testing. The other benchmarks were run, however, and very good results were achieved (see Figure 4 below.)

The processor itself seems to be RISC only in name when compared with the others here, as although it has a reduced, simple instruction set it is extremely complex in design. The main difference is that it is super-scalar, allowing it to use the integer maths unit, the floating point unit and the memory handling unit in parallel and, as such, could execute three instructions at once. This would require it to be fed an instruction each of the integer, floating point and load/save types in a row, but a compiler optimised for it could make that more likely than it might at first appear. Intel suggest a sustainable one and a half instructions per clock cycle. A small cache for 4K of instructions and 8K of data is provided on-chip, but this is too small to be of much use to Muse, and its presence means that board designers would be unlikely to include a larger secondary one. It also uses a three stage pipeline, and includes hardwired Z-buffering and some complex graphics instructions. The benchmarks showed that it was credibly fast at floating point operations and carrying out function calls (context switching).

Although all these technical achievements makes for a very fast and efficient piece of silicon, it also makes scaling the design far more difficult - a problem when this is the only possible design. One of the advantages of simply designed RISC processors comes from the physics of the components and lines placed on the chips surface. The smaller they get the less resistance they have, signals travel faster and have less distance to travel, and they dissipate less heat. This allows the processor to be clocked faster, as each signal can be expected to have reached all of the components before the next is sent, and there is less worry of heat damage or heat generated errors. The only limitation to this shrinking effect comes when the lines get so close together that electrons manage to "tunnel" between them. The MIPS and SPARC processors have both been able to use this to great effect - Sun quote a doubling in performance every 16 months so far and for the foreseeable future - but after nearly three years Intel have only just been able to announce a new version of the i860 that runs at 50Mhz. Further increases might be possible, but not with the ease of the simpler designs.

Intel was the first company to produce microprocessors, but the i860 was their first attempt at a RISC architecture, coming noticably later than their competitors. Although some machines have been based on it, it has been used mostly as an acceleration option, particularly in graphics systems. Intel has also produced the i960, with a smaller package and pin count, achieved by shaving off some functionality and cache size, aimed at the embedded systems market. At the time of benchmarking there were three i860 and four i960 board level products. Judging by the court action in late 1990 between Intel and AMD over the latter's 80386 compatible processors, we cannot expect to ever be able to second-source these processors.

11.2 The Motorola 88000

This processor was also tested as an accelerator board for an IBM PC compatible, which in this case was an Opus board running at 25Mhz. It had quite a small cache, but ran a System V Interface Definition (SVID) compliant implementation of UNIX, allowing easy implementation of a Muse port.

Motorola have designed a complete chip set for the 88000 series, including the 88100 processor and the 88200 cache/memory management unit (CMMU) which each incorporate 16K of on-chip cache. Up to four can be connected to each 88100. This results in a consistency of available products - all the board level implementations on the market at the time of testing were

configured with the same cache design. The design incorporates a three stage pipeline and "burst mode" memory read capabilities, which allow it to rapidly read a sequential group of memory locations from 'static column' dynamic RAM. The Opus board had two CMMUs.

The benchmarking showed it to be fast at carrying out function calls and on floating point calculations, but slow at referencing 16-bit data and pointer dereferencing. This last point is a major disadvantage for Muse execution. It also showed poor performance when comparing data types - being between four and six times slower than the others.

Like the SPARC and MIPS processor, the 88000 has the backing of a group of hardware and software manufacturers touting themselves as an "Open Systems" group. In this case it is *88Open*, who provide a specification and test for compliance of their standards. So far Data General is the main manufacturer beyond Motorola themselves to be delivering compliant machines on a large scale with its *Aviion* range, which does include competitively priced multi-processor systems that might provide a laboratory based development system for DMuse. Encore also uses them in their machines. The 88000 has not enjoyed much success and, since the agreement between Apple Computers Inc. and IBM Corp. in mid 1991 resulting in Motorola being brought in to build the single chip version of the RS/6000, its continued existence has to be questioned. Both Motorola and Tadpole produce 88000 based board level products, available in single and multi-processor configurations with real-time UNIX implementations. at least five other such systems also exist.

11.3 The AMD Am29000

The benchmarked system was an evaluation board, for which code had to be crossed-compiled on a host machine, running at 25 Mhz. Like the i860 cross-compiler, this was ANSI compliant, and Muse could not be ported for testing. The design included a very small cache. The benchmarks showed it to be slow at referencing 16-bit data, pointer dereferencing and particularly slow on floating point operations. A floating point coprocessor could be added, but pointer dereferencing is very important for Muse operation. It also showed it to be very slow at adding two numbers referred to by a double pointer dereference, which could be either a compiler fault or a failure in the data pipeline management.

The processor itself is designed for use in cheap embedded systems, and so uses slow memory. It makes up for for this by including facilities for burst mode accessing and a 512 byte internal branch target cache. It also contains a four stage pipeline.

The Am29000 is quite an old chip, and has seen more use in academic projects and embedded systems - including many laser printers - than in commercial workstations. It would therefore be quite difficult to produce a development system for Muse on such a platform, although at least three board level products exist which use cross-compilers and development tools on host systems. AMD themselves are probably better known for producing go-faster versions of Intel's 80x86 range, and general purpose PLDs, etc., than for this product, and its future is unknown.

11.4 The MIPS R2000

The R2000 was tested as part of a UNIX workstation, the MIPS RISCstation 2030. This version of UNIX was SVID compliant, and so porting Muse and the benchmarking code was quite a simple process. Being a workstation, it had quite a large cache - 32K data and 32K instruction, and the system was clocked at 16.67Mhz - the slowest tested. The processor also incorporates a five stage pipeline. This gave it quite an advantage in the normalisation calculations, and since its performance was so competitive anyway, suggests the likelihood of even better reslults with later generations that clock faster. The benchmarks showed it to be fast at floating point operations (except floating point division - a problem with the Floating Point Unit (FPU)), array referencing and local data access. The results showed it to be fast at single pointer dereference but not so fast with double pointer dereference, which suggests either that the compiler doesn't track double pointer references, or that the pipeline stalls on a pointer to a pointer. These are important to Muse due to the structure of its symbolic data frames.

At the time of the benchmarking the CES RAID 8235 board level system was about to be released, based on an R3000 with a real-time kernel and development support from a Sun workstation. Other such products are also available.

The original MIPS processor was the first commercial product to use the academically proven performance advantages of the RISC architecture. At the time of writing this has been much improved upon, and has been chosen as the standard processor for a range of personal computers/workstations backed by Digital Equipment Corporation, MIPS themselves, and a large number of IBM PC compatible hardware and software manufacturers, including Compaq calling themselves ACE. However, Digital's requirements for their own system compatibility mean that MIPS have to produce processors with the word order reversed - small-endian rather than the usual big-endian - for them, leaving the question of compatibility in question. Also, *The Apache Group* has been set up to counter ACE by AT&T and Unix International. They have started MIPS/Open to standardise on MIPS with Unix SVR4 instead of the more PC oriented designs of the ACE consortium.

If nothing else then, this suggests a continued life for the processor family, and that a lot of effort will be put into producing a range of products that might be used as DMuse platforms - the R6000 series processors should be in production within our timescale offering far greater performance.

11.5 The SPARC

The benchmarked system was a Sun SPARCstation 1, which ran a 20MHz clock, producing approximately 12.5 SPECmarks. The benchmarks showed it to be slow at integer multiplication - which is used heavily in array referencing. The SPARC uses a software library function for this while the other processors (except the Am29K) do it in hardware. However, it did not prove to be as slow at integer division, while the FPU was unusually slow on floating point division. The result for *sprintf* was also extremely slow at the optimisation level used, and this skewed the SPARC average timings.

The first thing to be noted about the SPARC (Scalable Processor ARChitecture) design is that it is a book - an open specification which can be purchased for the price of its printing and used to produce compliant processors. The only proviso that Sun places on its licensing is that they have first access to any new version. Due to this, there are now over eight chip manufacturers working on producing SPARC compliant implementations - each putting their own technological advantages to use and competing with each other, rather than producing for the slightly more "niche" marketplace of the general RISC processor. This means that SPARCs are gaining in performance faster than the others, and bringing all the other advantages of multi-sourcable products, such as pricing and availability. Also, as the SPARC standard is increasingly taking over the RISC workstation market, the

number of products - both hardware and software - available is also increasing. Many of the companies who have made their names making IBM PC clones are now working on SPARC clones - most notably Opus and Compuadd. This gives us a wider choice of platform and a wider market for the final development system.

The SPARC specification is for an integer processor system only, with floating point work being carried out by an external FPU. Different FPUs, including some by Weitek, can be connected for various levels of performance. It is possible to connect more than one, and some of the more recent implementations are designed with on-chip FPUs, fixing the type used but reducing inter-communication times. The SPARC compliance manual includes definitions of the memory bus, the S-BUS expansion connectors, and the methods of inter-processor communications. There are no regulations on the shape, pinout or manufacturing process, allowing products to be developed using the most suitable packaging and substrate materials. Gallium-arsonide and bipolar chips are being developed. The specification defines cache size and a register window system, the last of which is useful for the Muse virtual machine model. At the time of testing board level implementations - specifically the Sun SPARCengine 1E, was based on this system, but an enhanced SPARCengine 2E is now available, capable of over twice the performance. Within the timescale of this project it is expected that board level products based on the new chip sets from Tera Microsystems and/or Fujitsu Microelectronics embedded high performance processors will also become available.

At the time of writing, Solbourne and ICL were selling inexpensive symmetric multi-processor workstations capable of running a DMuse implementation, each running their own SUNOS superset to control the load balancing. Sun themselves were discussing a system using four *Viking* processors in a workstation box with their own implementation of a load balancing SUNOS, which they expect to be available in 1992. The Viking processor is a 40Mhz, superscalar, BiCMOS implementation being built by Texas Instruments that has been rated at anything between 50 and 80 MIPS, depending on the usual superscalar compiler techniques. Unix International, the group led by AT&T and Sun in developing Unix System V Release 4 (SVR4), have stated that the multi-processor version of SVR4 will not be available until '92/93. Recently the press has commented that Sun's first multi-processor systems, the *Galaxy* range, will only use asymmetric multiprocessing implemented with a maximum of two pairs of 40Mhz Cypress SPARC implementations. It is believed that the Viking based systems will be called *"Jupiter"* and is currently waiting for the completion of the processors and a working operating system to run under. Sun are believed to be having a lot of trouble producing a symmetric multiprocessing version of SVR4, but hope to release it as part of their *Solaris 2.0* system, due in the middle of 1992. This hardware/software platform might make a useful laboratory based DMuse development system.

11.6 Others

It should be noted that since these tests were carried out other processors have been released. These include the IBM RS/6000 chip set and the HP/Apollo "Snake". The former could not really be considered here anyway - it is a RISC in name alone, needing a set of five chips to form a working system. However, IBM have recently announced that they have managed to mount nine chips at substrate level onto a single further substrate which contains all the necessary interconnects and reduces the size and pinout problems. This should allow them, and Motorola, who have been named as a co-manufacturer of this part, to start producing smaller and cheaper single package processors that might be viable for DMuse in a year or two. IBM's recent joining of forces with Apple Computers Inc. suggests that some inexpensive development systems might become available from one or both soon after. This will probably be too late to be worth evaluating for the first DMuse, however. The HP/Apollo processor, nicknamed "Snake", is available now in a range of workstations, and the published performance figures are very impressive. These workstations run a version of UNIX - HP are Open Software Foundation (OSF) members - and it can be assumed that multi-processor versions will be available soon if not now. It will have to be taken into account when a final decision is made.

12. PROCESSOR BENCHMARKING

The following results were produced. It should be remembered that the performance figures given are as a percentage of the 68020 execution time normalised for clock rate - the smaller the better. The estimated Muse figures are derived from the Heines and Jalics results, and so their accuracy is compromised by the problems mentioned above (see especially the 88000 results.)

Processor	Clock Rate	Heines & Jalics	Muse Execution	Estimated Muse
68020	20.00	100.00	100.00	100.00
SPARC	20.00	39.50	38.02	33.75
R2000	16.67	26.03	27.16	24.91
i860	33.00	33.40	-	27.92
88000	25.00	33.41	52.25	33.34
Am29K	25.00	40.32	-	38.37

Figure 4: Summary of RISC Benchmarks

Although the estimated Muse figure seems quite accurate for the SPARC and MIPS R2000 processors, the figure calculated for the 88000 shows that we can't altogether rely on the estimated figures for the i860 and Am29K. Of the actual Muse execution tests, we can see that the MIPS was fastest, then the SPARC and slowest was the 88000. The Am29K is definitely the slowest overall, and we see no reason why it might run Muse any faster than estimated. We can, however, see some reasons why the i860 might actually run Muse slower than suggested by the Heines and Jalics tests. The processors were therefore ranked thus:

(1) MIPS R2000
(2) Intel i860
(3) Sun SPARC
(4) Motorola 88000
(5) AMD Am29K

13. COMPILER BENCHMARKING

As some RISC manufacturers claim that compiler optimisation is an important part of performance, some testing was undertaken to compare optimisation techniques for each processor. Figure 5 shows the resulting percentage speedup on unoptimised code portions. Since in many cases an adverse effect was seen, both the best and worst test results from the Heines and Jalics sub-groups are given to indicate how much risk there is in using the compilers. These were the bundled Sun SPARC C compiler; The MIPS C compiler; the Green Hills C compiler for the i860

and the Diab Compiler for the 88000. No such optimisation was available from the Am29K compiler. As can be seen, the compilers that can most dramatically increase code execution speed can also decrease it considerably if used indiscriminately. Usually the same optimisation techniques that would be particularly beneficial to Muse caused bad floating point results from the MIPS and Sun compilers. However, it should be noted that the Sun compiler achieved some very impressive results with optimisations, and the new unbundled Sun C compiler is said to produce far better optimised code than the old bundled product.

Processor	Heines & Jalics		Muse
	best %	worst %	Execution
SPARC	68.14	-15.39	45.04
R2000	57.66	-122.67	25.52
i860	21.63	-27.82	-
88000	22.26	3.41	12.26

Figure 5: Percentage Speedup from Optimisiation

14. PROCESSOR SELECTION

Other factors had to be taken into account when finally selecting the processor to be used as a platform for DMuse. Many of these factors are still varying, and we will not make a final decision until we are actually ready to make a purchase and start implementation. Some of the other variables relevant here are:

- The development system is seen as a major part of the DMuse system. Without a good set of tools to build and debug multi-threaded real-time KBS systems their construction might be too laborious to be worthwhile.
- The level of effort necessary to convert Muse to the new platform and build in the new enhancements must be cost effective for DMuse to be an acceptable proposal. This means taking into account the availability of Muse on each architecture as well as ease of migration and hardware construction and the development tools available.
- The operating systems/kernels available on the processors for our use. The development system would be best running a flavour of UNIX - preferably a major open standard one - for ease of migration and use. Only the MIPS and SPARC machines ran a version of UNIX which supported the Berkeley Standard Distribution (BSD) socket system which is used for inter-process communication in Muse, for example. OS support for multi-processors would simplify their use in implementing and load-balancing separate Muse tasks. However an embedded/flyable final delivery system, which we must keep in mind as the final target of DMuse, must be capable of true real-time operation - something for which UNIX is not currently famed. Therefore a good and as similar as possible real-time kernel should also be available for the processor.
- Although we are looking for the fastest possible processor, we must keep in mind the problems we will face when this hardware is finally implemented on board an aircraft. High clock speeds are fine in the laboratory, but this would not be true in the electro-magnetically noisy environments they will be placed in - especially in helicopters - as higher frequency circuits are more susceptible to radiated interference. We must therefore look for processors that get as much done for as little clocking as possible to counteract this - in other words super-scalar designs as seen in current MIPS, i860 and SPARC designs. We have yet to discover what this limiting factor is.

At the time of the benchmarks the final choice of processor was left open, but at time of writing the SPARC processor seems to be the best choice for the development of a lab. based DMuse system. Muse is already available, and highly stable, on this platform, making it the most cost effective solution. The performance figures quoted on new versions, as they become available, closes the performance gap with the current Intel and MIPS offerings, and thereby reduces any compromises we might be making on price/performance. The SPARC currently seems to provides the best development and target platform, simplifying the use of DMuse when finished as well as in its production. Also, as discussed below, there is good availability of multi-processor workstations with multi-process/threading control built into the operating system at costs and time scales within the range expected for the DMuse contract.

15. TARGET MACHINE SELECTION

Three possible architectures became apparent to us as we approached this side of the problem. Our previous experience with Muse had at first been on workstations, and recently a project entered into with GEC Avionics Flight Automation Research Labs (FARL) was made to deliver acceptable speed by spreading the load over multiple workstations on an Ethernet network. We also had a previous project transferred onto a VME based single processor crate which was strengthened to allow it to be flown on board a Lynx helicopter owned by the then Mission Management department of RAE - now DRA Flight Systems. The third option described below came to us late on in the original study contract, when multiprocessor workstations using some of the processors tested above came onto the market. None of the three would cause great problems in implementation of DMuse, and so the only real values to be balanced are those of cost, performance and adaptability / upgradability.

15.1 Loosely Coupled Workstations

The work with GEC on the Intelligent Displays Manager for fixed wing aircraft was originally developed on Sun-3 machines, and to maintain the necessary performance as the system grew in complexity it became necessary to separate out knowledge sources and graphics software and allow them to communicate through harness processes written in 'C' which passed messages between them using Ethernet. Initially the largest knowledge base was placed on one machine and the second, smaller knowledge base and the graphics software were put on a second. As the system grew it later became necessary to move the graphics onto a third machine, but this was accomplished easily by altering the harness structure in 'C', showing the inherent upgradability of such a model. The only limit on expansion is the theoretical limit to the number of active machines on the network.

However, although developing DMuse to run on such a system of interconnected workstations (see Figure 7) would provide the most cost effective hardware solution, it would be a great performance compromise, since Ethernet, at 10Mbits/second, is nowhere near as fast as some of the inter-processor buses especially designed for such data interchange. It would also be the most bulky (each being separately cased) and most prone to fault of the three systems. Finally, such a system would enforce a local memory model on us, as each processor would be a separate entity, and this was not felt to be acceptable.

15.2 VME based Processor Crate

A system based on a VME backplane looked the most likely at the time of the original feasibility study, as it offered the option of building either a laboratory deskside unit connected to a host or a hardened flyable crate to be fitted inside an aircraft without

having to greatly change the overall design. Using a system of slot-in cards would also allow us to decide on how many processors to include, and to select card designs with on-board memory, or which shared a separate plug in memory card in the crate, or a mixture of both (see Figure 7.) Although VME was designed for fast bus communication it is no longer considered as "fast" due to technology's moving edge. However, some other bus design could be used should we go ahead with this model.

At the time of writing this is still a viable option, and we are keeping an eye on the board level products available as the market grows. Many of the products examined in the feasibility study provided some hardware support for multiple processors, the most common being dual-ported memory shared between the CPU and the VME bus. The only limitation that has been noted is that as new models of processors become available, offering greater performance, they initially get built into desktop systems and only some time later make it out as embedded versions. Crates of variable sizes are available to allow different numbers of slots, and some boards are becoming available which contain multiple processors on them.

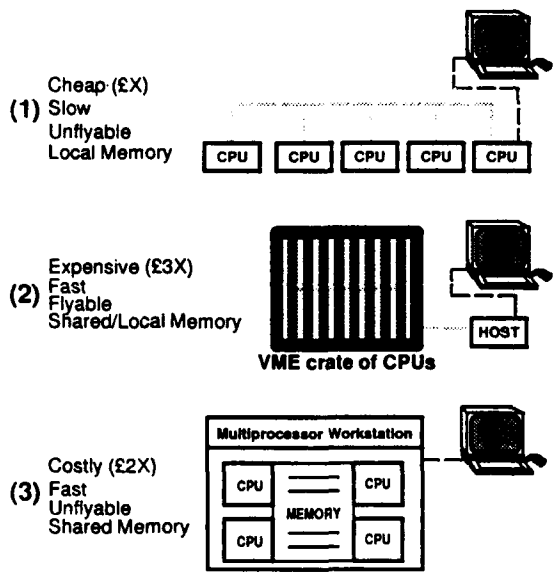


Figure 7: Possible Hardware Configurations

15.3 Specialised Multiprocessor Workstation

Over the last couple of years various companies have started to release workstations using two, four or even eight processors communicating in the most part via specially designed high-speed buses using the fastest available parts. Silicon Graphics offer a large machine containing multiple MIPS processors, and Solbourne, ICL, FPS and now Sun offer machines based on multiple SPARCs. Sun's *Jupiter* machine mentioned earlier, if it matches expectations upon release, will provide probably the best performance results for this type of machine to date. The figure of four processors may be too limiting, however, and resorting to network communications between multiple such machines, although possible, might become a bottle-neck unless a fast network is utilised. Due to the operating system support for inter-process communications not differentiating between where those processes reside physically, the design of DMuse is unlikely to show any difference to the developer when communications are being made over inter-processor buses or networks. While this allows us to build a system this way and still give a standard model to the programmer, it also doesn't give them any hints as to possible performance bottle-necks in their design.

Such a system would probably be built with shared memory but local cache, allowing either memory model to be used in DMuse. All the buses used in these designs are notably faster than a VME bus, being in most cases built at board level with sockets for piggy back processor boards to be plugged in. Both Sun and Solbourne have designed buses with far greater possible communications bandwidth than they can currently possibly use and allow the plugged in processors to run at their own speed, allowing these processors to be upgraded by simply swapping them with faster units on the same bus.

The ICL DRS6000 is not currently SPARC compliant, having started with SVR4 - a version of which they developed for Unix International. They are apparently working on an updated workstation which uses more than four processors.

This final option also proves to be the most acceptable when viewing DMuse as a product for sale to other interested groups. By building DMuse for use on a commercially available series of machines, rather than providing both hardware and software, the system becomes far more attractive. Costs of such a system have not yet been finalised, but we believe that such a machine with local disk and console to make it a stand-alone system, would fit well into our hardware purchasing budget for the project. Overall, such a design, although it has some problems, is seen as the most favourable.

16. OPERATING SYSTEM SUPPORT

Sun has defined a multi-thread architecture for SunOS [12] which will probably be the main basis of their Solaris 2.0 operating system, incorporating SVR4 and a large number of Sun proprietary development and productivity tools, due for release in mid-'92. This would be the largest and best developed and supported operating system for DMuse as a development system for laboratory use, but may not be the solution for embedded use. Similar operating systems exist for MIPS multiprocessor workstations. The problem with all of these, however, comes when trying to run real time systems under Unix. No matter how good a system, it can't promise maximum response times when it doesn't know if Unix is going to be executing its code, or some other process, when an interrupt comes in. Some versions of Unix, however, offer real-time facilities. Muse also has the capability to be downloaded and run on a target machine without Unix, and it would be necessary to provide a similar capability to DMuse. The feasibility and study contracts therefore looked into various real time kernel systems and Unix implementations to derive some idea of the software support available. The major real time kernels provide explicit support for multiple processor systems, ranging from transparent allocation of tasks between processors to an implementation of TCP/IP that runs over a VME backplane.

The products checked were: LynxOS, TP-IX, VxWorks, pSOS, VRTX, MTOS, PDOS, dbxWorks and XRAY. Of these, only VxWorks and dbxWorks supported the SPARC and the Intel i960, while VRTX was being ported to them. None of them supported the MIPS processor at that time, although LynxOS, VxWorks, pSOS and VRTX apparently promised this in the future. The Am29K was only supported by VRTX while the Motorola 88000 was supported by, or soon to be supported by, all but dbxWorks. Again, this is a volatile marketplace, and beyond confirming that such things are possible, it is pointless attempting to make decisions on this level of product until we are close to production.

17. CONCLUSION

This paper has shown how the original definition and construction of the Muse real time IKBS toolkit, and it's use, has led us to define a system able to meet the requirements of in-flight Mission Management. We had learnt from previous experience that the original Muse had the necessary capabilities as a software tool, and we are now moving towards meeting the hardware requirements of a large scale MM problem.

As can be seen from our previous discussions of the results given here, it is almost impossible to draw precise conclusions in an area as volatile as the RISC processor marketplace. However, it is hoped that this paper has shown how we approached the problem of designing a real time knowledge based systems development environment from the groundwork already laid by previous research with Muse, and that a feel for the architectural possibilities, in both hardware and software, for such a system has been given.

18. REFERENCES

[1] Ching-Chi Hsu, Shao-Ming Wu, Jan-Jan Wu, "A Distributed Approach for Inferencing in Production Systems", p.62 Proceedings IJCAI 1987.

[2] Akinori Yonezawa, Etsuya Shibayama, Toshihiro Takada, Yasuaki Honda, "Modelling and Programming in an Object-Oriented Concurrent Language ABCL/1", Object-Oriented Concurrent Programming, edited by Akinori Yonezawa and Mario Tokoro, MIT Press.

[3] Yasuhiko Yokote, Mario Tokoro, "Concurrent Programming in Concurrent Smalltalk", Object-Oriented Concurrent Programming, edited by Akinori Yonezawa and Mario Tokoro, MIT Press.

[4] Dick Pountain, "Rekursiv: An Object-Oriented CPU", BYTE Magazine, November, 1988.

[5] V.R. Lesser, D.D. Corkill, "Functionally Accurate, Cooperative Distributed Systems", in Readings from Distributed Artificial Intelligence, edited by A. H. Bond and L. Gasser, pages 295-310.

[6] R.G. Smith, "The Contract Net Protocol: High Level Communication and Control in a Distributed Problem Solver", IEEE Transaction SMC-10, no.12, 1980.

[7] M. Bell, M.E. Bennett, R. Zanconato, CCL Report C1924-FR-001a.

[8] S.E. Conry, R.A. Meyer, V.R. Lesser, "Multistage Negotiation in Distributed Planning", in Readings from Distributed Artificial Intelligence, edited by A. H. Bond and L. Gasser, pages 367-384.

[9] E.H. Durfee, V.R. Lesser, D.D. Corkill, "Coherent Cooperation Among Communicating Problem Solvers", IEEE Transactions on Computers Vol C-36, no. 11, 1987.

[10] L.D. Erman, F. Hayes-Roth, V.R. Lesser, D. R. Reddy, "The Hearsay-II Speech Understanding System: Integrating Knowledge to Resolve Uncertainty", Computing Surveys, vol 12, pp213-253, 1980.

[11] A. Blackwell, "Benchmarking RISC processors for Muse" CCL Technical Report C3435-TM-005a.

[12] A. Blackwell, R. Zanconato, R. Smith, "The Provision of Parallel Processing Enhancements to Muse" CCL Final Report C3435-FR-001a.

[13] M.L. Powell, S.R. Powell, S. Barton, D. Shah, D. Stein, M. Weeks, Sun Microsystems Inc., "SunOS Multi-thread Architecture", USENIX Paper, Winter'91, Dallas TX.

IMPLEMENTATION AND OPERATIONAL EXPERIENCE WITH A NEW ARRIVAL TRAFFIC MANAGEMENT SYSTEM AT THE FRANKFURT ATC-CENTER

M. Schubert
U. Völckers
German Aerospace Research
Establishment (DLR)
Institute of Flight Guidance
Flughafen
D-3300 Braunschweig
Germany

Summary

The DLR-Institute for Flight Guidance had developed a new Arrival Traffic Management System called COMPAS (Computer Oriented Metering Planning Advisory System).

The system has been implemented at the Frankfurt Regional Air Traffic Control Center in mid 1989 and is in continuous operation since then.

A comprehensive implementation plan was developed in order to guarantee a smooth introduction in particular with respect to both technical issues and controller acceptance. The experience gained and the lessons learned during the implementation process as well as the operational results will be reported.

1. Introduction

The job of the arrival controller is to transform a random input of various types of aircraft, from various directions into an orderly sequence of properly separated and spaced aircraft feeding into the final approach path. During the last years, DLR (German Aerospace Research Establishment, Institute of Flight Guidance) has developed, tested and implemented a metering, spacing and sequencing function for the planning and control of the arrival traffic. It is significant for COMPAS, that the computer provides advisories how to handle each aircraft in the system. The controller is still in charge of the traffic situation, and is free to disregard or modify the suggested plan.

The computer serves as an aid to judgement, but the functions do not enter the primary control loop (figure 1).

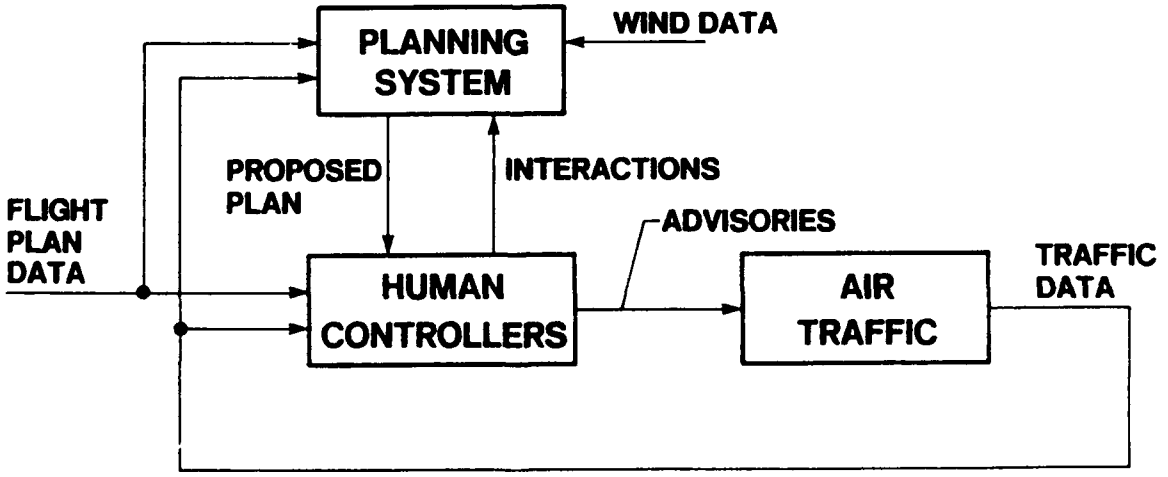


Figure 1 Planning and executive functions

The system has reached a particular level of automation and complexity especially for the functions landing time prediction, sequencing and spacing, flight control, human-machine-interface and integration within the overall air traffic management system.

A solution, where the computer system works fully automatic, with the human controllers only monitoring the system, in order to take over in case of system failure, presently is for many reasons neither practicable nor acceptable.

That is the reason why so far the degree of automation is guided by the philosophy of giving computer assistance to the human controllers. The system releaves the controllers of routine functions and supports them with all necessary data to take safe, efficient and orderly control actions. The decision making itself however and thus the overall responsibility is still left to the controllers.

Because the theoretical background of COMPAS already has been described before [1, 2], there will be no further discussion of the basic COMPAS algorithm. Basically, COMPAS is a computer controlled planning system (figure 2) which helps the ATC controllers to plan and control the inbound flow of air traffic into the terminal control area (TMA) of an airport more efficiently. The planning algorithm suggests a landing sequence and schedule to air traffic controllers taking into account a variety of constraints and parameters in order to exploit the available capacity of an airport in an optimal way. Unnecessary delays will be avoided and unavoidable delays will be absorbed as early as possible. The basic ideas were tested with an experimental COMPAS in a simulated ATC environment and the results are reported in [3]. After the implemention of the operational version of COMPAS a second evaluation was carried out and is reported in [4].

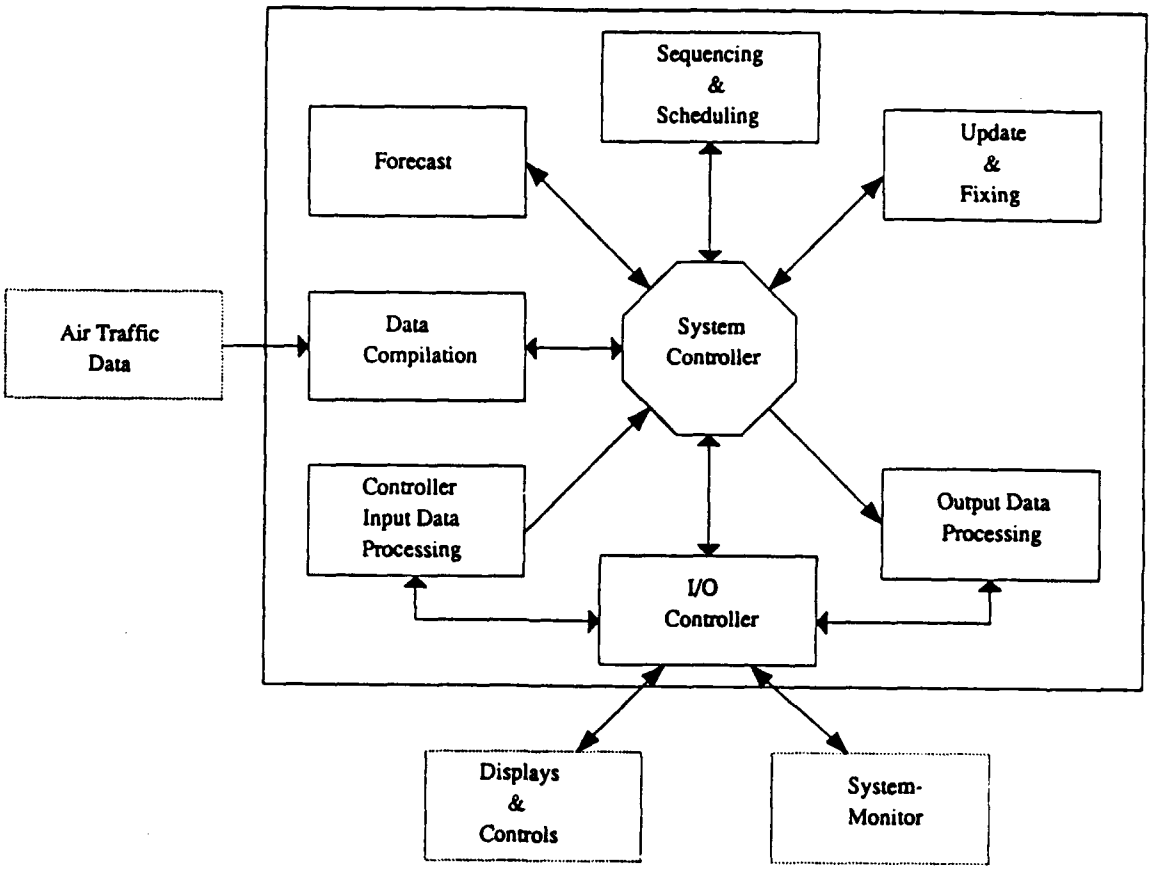


Figure 2 COMPAS data processing functions

2. Objective of the project COMPAS-OP

Based on the positive results of the experimental studies of COMPAS obtained in the environment of the Air Traffic Management and Operations Simulator (ATMOS) of DLR the project COMPAS-OP was defined in order to develop and build a prototype system for the ATC center at Frankfurt/Main (figure 3) and to test and to assess this operational system under real ATC conditions. The mathematical planning functions and the human interface of the experimental system, which were already optimized in the simulator studies, were used as a starting point.

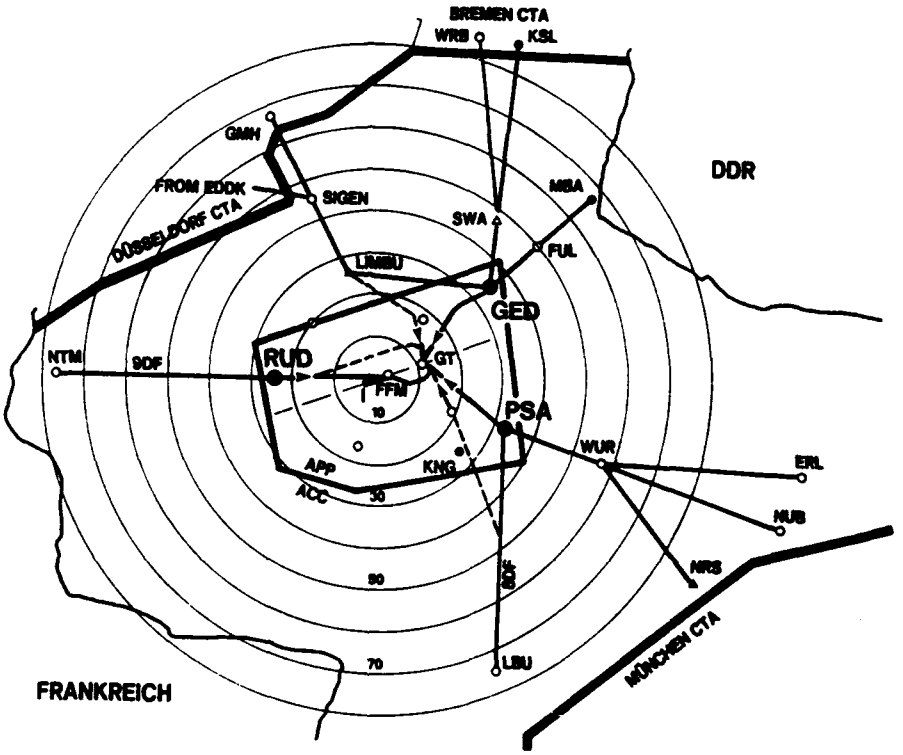


Figure 3 Frankfurt/Main airspace structure

The operational COMPAS system (COMPAS-OP) was designed for Frankfurt/Main ATC environment. Eight controller positions, six for the Area Control Center and two for the Approach Control have been equipped in 1989 with COMPAS displays and keyboards. During the first half a year of operation, the system was evaluated under real ATC conditions. Special software equipment for data recording and evaluation had been integrated into the COMPAS-OP main computer [4].

Due to the prototype character of the current system the main computer and the supporting subsystems in the starting phase were not redundant. In 1990 it was the task of an industrial manufacturer to build a redundant system according to the specifications of the BFS. For the construction of the prototype system only off-the-shelf hardware was used. Similarly the software development relied on standard software libraries readily available.

Although an evaluation of the functional capacity and the operational performance of the planning system remains the overall objective, a field test of an operational system will be necessary in many ways different from a simulator evaluation made at DLR.

The simulator evaluation of the experimental system dealt with the feasibility of computerized approach planning in general. It focussed on the comparison to conventional

procedures, thereby making use of the exact reduplication of traffic scenarios which is only given in simulation.

Real-world operation does not offer this feature. Here, the opportunity of any comparisons between COMPAS-assisted and not COMPAS-assisted control is much more limited. In the field test evaluation, the focus was much more on verifying the simulation results and generalizing them in terms of

- including all working positions in the control center which are involved in handling approaching aircraft,

- handling all traffic situations that can occur in real operation, and

- observations based on long-term operation of the system.

The field test evaluation should make use of the three main data sources: traffic data, controller activity and controller judgements.

3. Technical implementation of COMPAS-OP

The following functions are essential to the COMPAS and are processed with a cycle time of 4 seconds corresponding to the update rate of the radar position data of the aircraft responses:

- Reception of the radar data packets from the BFS DERD-MC-System, transformation into an appropriate coordinate system, identification of the aircraft data by means of the available flight plan data base, handling of the identified aircraft parameters,

- continuous speed calculation of the identified aircraft using the radar coordinates,

- identification of the route to the gate by means of the flight plan data (ZKSD-System), calculation of the time the aircraft is passing the metering fix and the gate,

- calculation of the flight path and the flight time on the basis of the aircraft performance data and the parameters of the selected Standard Arrival Route (STAR),

- computation of the COMPAS planning algorithm giving the sequence of aircraft, update of the suggested COMPAS times over the metering fix and gate, conversion of the time updates into advisories to the ATC controllers,

- display of the computed sequence of aircraft and advisories on the ATC controller working positions.

Some other functions are run once or on request only:

- Initialization and configuration of the COMPAS system by means of data sets describing the airport and airspace environment and the aircraft characteristics,

- aquisition of weather information and flight plans which are received about 20 minutes before the aircraft enters the TMA,

- processing of the ATC controller keyboard inputs as part of the concept of local interaction and processing of resulting COMPAS plan modifications.

Some other functions are included in the system, too, which are, however, not part of the kernel software. They include data aquisition, data processing and quick-look functions.

The sum of all these functions forms the basic part of the operational COMPAS. The experimental version of COMPAS included these functions, too, but some simplifications were tolerated because of the limited number of working positions and the application of standard interface components. In addition the greater number and the local accomodation of the displays and keyboards, the interface to the ATC computers employing appropriate protocol specifications and a reduced cyle time compared to the experimental system had to be taken into account in converting the experimental into the operational system. The specifications and restrictions presented led to a complete reorganisation of the data processing concept. The single computer solution was discarded and a distributed data processing system was selected in which the functions interface to the ATC system, protocol conversion, pre-processing of the ATC data, the COMPAS planning algorithm and the human interface are run on separate computers linked by a Local Area Network (LAN) (figure 4).

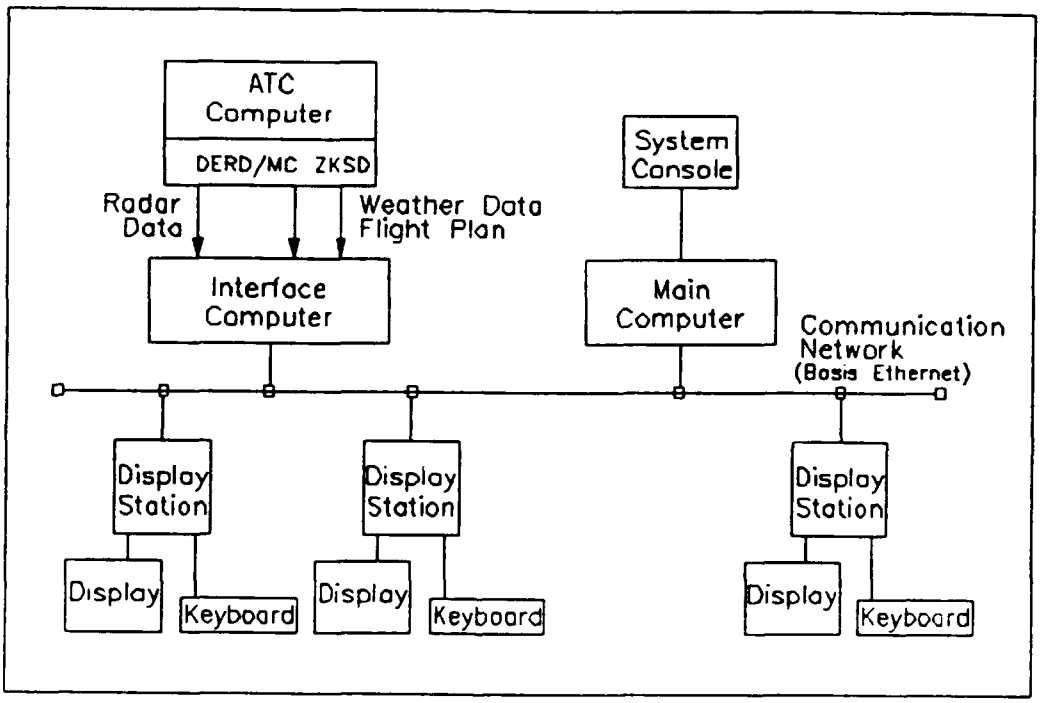


Figure 4 COMPAS-OP processing system

The functions data aquisition and data pre-processing are assembled into the Interface-Computer (SR). All the functions to interface the system with the ATC computers are concentrated here. The protocol handling, the pre-processing of the received ATC data and some form of data conversion are done by this computer. For the physical interface of the radar data and the processing on the lowest level of the protocol specification, an intelligent interface was built which is part of the Interface-Computer and which reduces its processing load. Another interface is used for the communication of COMPAS with the central ATC flight plan information and distribution system (ZKSD, DÜV) which makes available the flight plan and weather information.

The COMPAS Main-Computer (CR) receives the pre-processed data from the Interface-Computer and runs the COMPAS planning algorithm. The generated sequence of aircraft and advisory information are then transmitted to the Displaystation-Computers to display the appropriate sequence of aircraft to the ATC controllers. Requested modifications of the planned sequence are entered by the controller by means of the function keyboard and are passed back to the Main-Computer which takes into account this information in the next cycle of the COMPAS planning algorithm.

The display and input functions are assigned to Displaystation-Computers (AR) which get their information from the COMPAS Main-Computer. Each ATC controller position to be equipped with the COMPAS system has it's own computer, a display and a function keyboard. The Displaystation-Computers are configured according to the type of ATC controller working position.

All computers are linked to each other by a Local Area Network (LAN). The LAN provides the necessary communication to distribute the information among the computers and allows a local separation of the individual units.

The data flow which results from the hardware concept of the system is shown in figure 2.

The advantages of the implemented system structure are:

- Improved processing power due to the distribution of the required functions on more than one computer,
- separation and modularization of groups of functions,
- more efficient integration and test of subsystems,
- local distribution and separation of the ATC controller display stations,
- ability to expand the system in particular to a greater number of ATC controller working positions if needed.

The display and keyboard subsystem is an important part of COMPAS because the human interface components are concentrated here. In the experimental COMPAS the display and keyboard layout was investigated, tested and optimized. Only minor modifications had to be made for the operational COMPAS. A graphic subsystem is used to display the planned sequence of aircraft in 16 colors with a resolution of 756 x 512 pixels (figure 5). For the interaction with COMPAS an optimized small function keyboard is used (figure 6), which is linked to the Displaystation-Computer by a serial data line. Some of the interaction sequences are handled locally in order to reduce the overhead demands on the communication link and the Main-Computer.

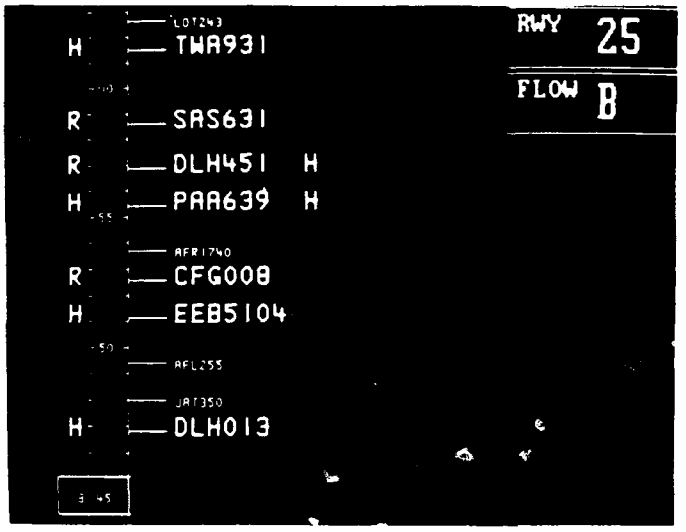


Figure 5 Display of planned aircraft

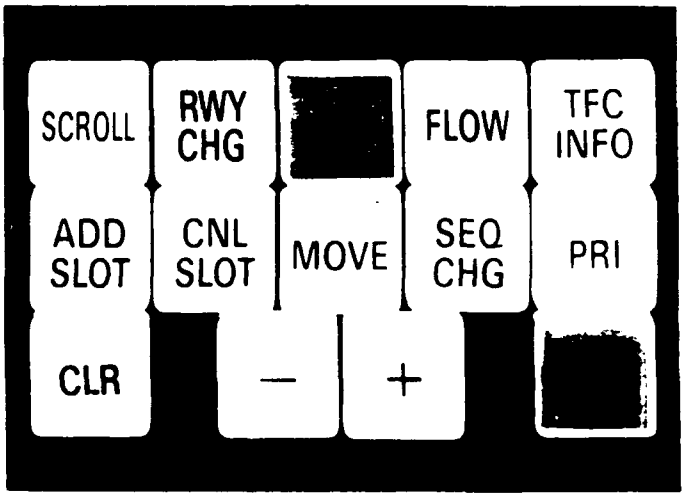


Figure 6 COMPAS function keyboard

The given specification to handle 70 planned and controlled aircraft approaching the airport at the same time was verified in the simulation after the integration and the first test of the COMPAS subsystems. For this purpose the Air Traffic Management and Operations Simulator (ATMOS) of DLR was complemented by simulated DERD-MC and ZKSD functions and interfaces. The tests were run satisfactorily at peak traffic loads and all COMPAS system components (DERD-MC interface, Interface-Computer and Main-Computer) had sufficient spare capacity even to implement future expansions of COMPAS. In the tests special operational conditions were simulated, too. They included radar dropouts, aircraft crossing radar receiving areas, the duplication of code allocation and incorrect inputs of callsigns by the ATC controllers (e.g. LH123 rather than DLH123).

The installation of the system at the ATC center of the Frankfurt airport was started in the first week of April in 1989. A detailed planning of all activities was necessary to avoid disturbances of the ongoing ATC operations. After the training of all 400 controllers and the servicing staff was completed the displays and keybords were placed at their final positions within 2 days. On 18th of Semptember 1989 COMPAS was put into operation at the Frankfurt ATC center. Figure 7 shows the approach position with the COMPAS display located on the lefthand side of the radar display and on top of the Weather Information Display.

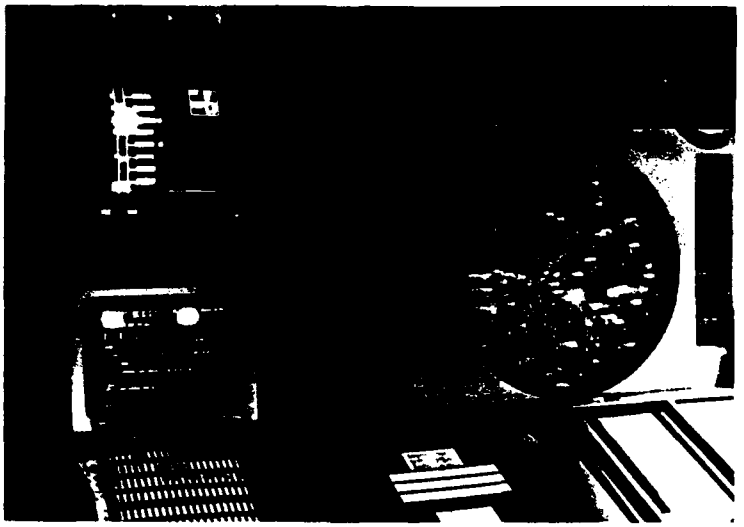


Figure 7 COMPAS display in the ATC APPROACH of Frankfurt

4. Results of the Field Test Evaluation

For a detailed description, it is helpful to distinguish between results which apply to APPROACH control, to ACC (Area Control Center) control, and to approach traffic as a whole.

APPROACH Control

Seeing that COMPAS makes an overall plan which aims at an easy flow of the approach traffic, thereby integrating all areas and functional control units involved, a central parameter of traffic flow or landing rate, respectively, needs to be specified. COMPAS always plans for a preselected traffic flow rate which can be varied manually by APPROACH control due to numerous factors (wheather, runway condition, departures, etc.).

This is done by using the FLOW input function of the system to select a minimum separation in Nautical Miles (NM) applying to aircraft on final approach.

So the selected FLOW becomes a central parameter, which has a direct impact on ACC control, too, because it determines immediately the maximum rate of aircraft that can be handed over to APPROACH control at the TMA boundary.

The examples given in figure 8 and figure 9 shall emphasize the importance of this parameter. For two days they show at the left hand part of the figures, over the time axis,

- the FLOW as specified in Nautical Miles minimum separation,
- the number of aircraft which, by COMPAS calculation, theoretically could have passed the Gate if there were no other traffic, but actually have not ("holding aircraft"), and
- some traffic statistics per 30-min. interval: landing frequency is displayed as a letter histogram, with letters referring to the classification of heavy/medium/light aircraft (obtained from actual time of arrival, ATA). The number of intended landings is displayed as a column (obtained from calculated time of arrival, CTA).

Horizontal radar tracks of the two-hours traffic peak from 09.30 to 11.30 are given at the right.

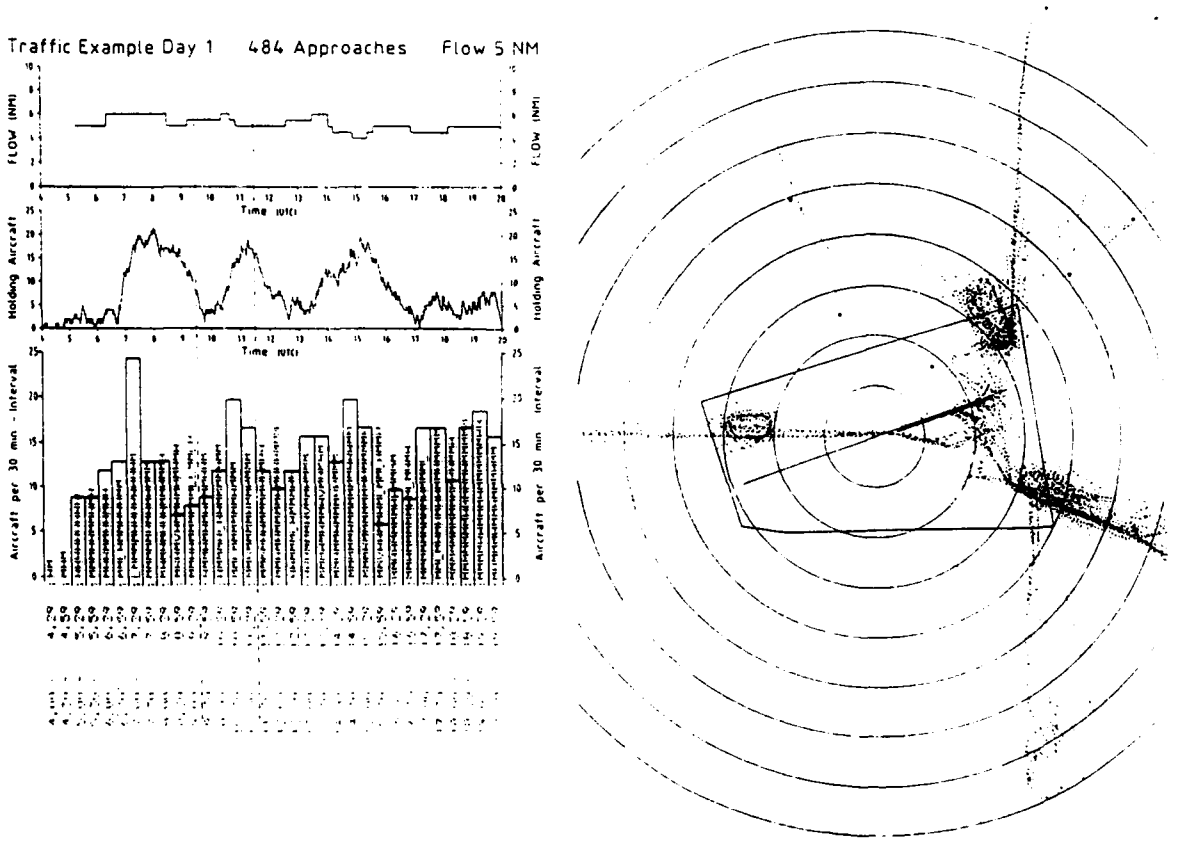


Figure 8 Traffic flow example with selected FLOW of about 5 NM (see text)

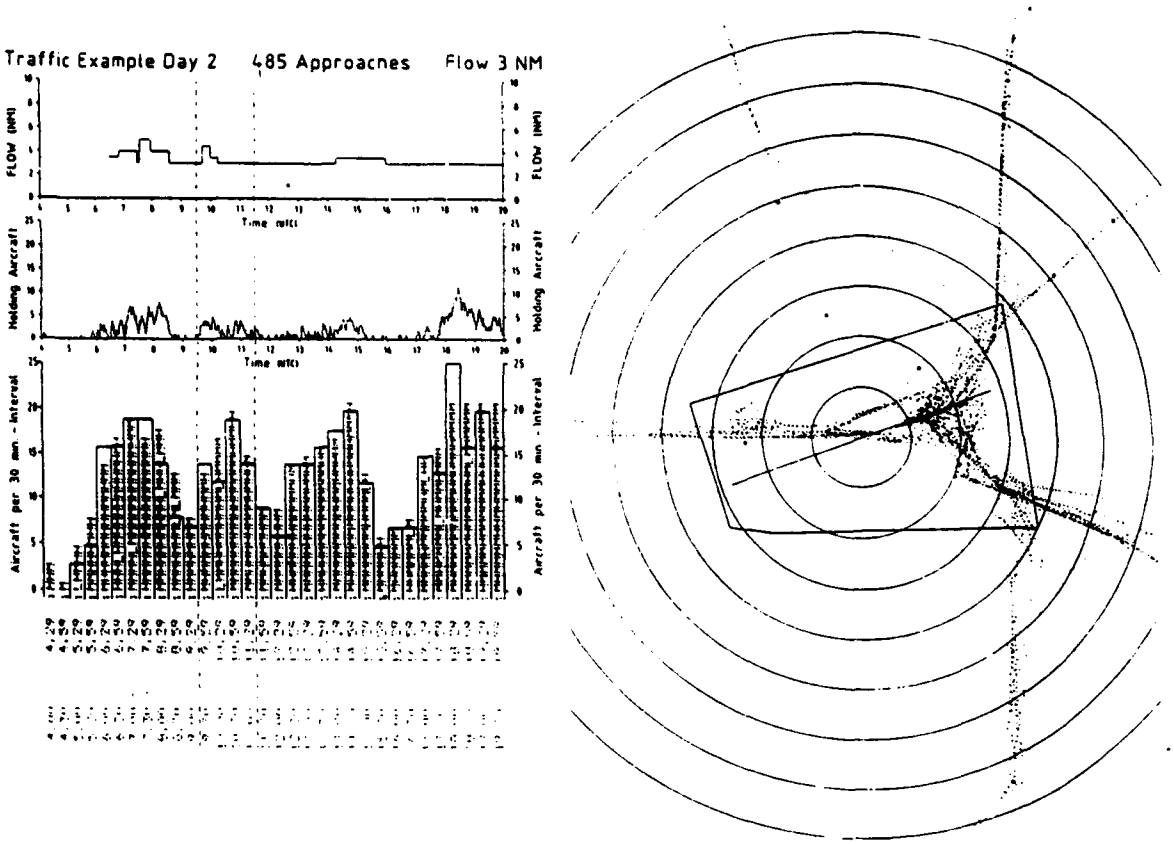


Figure 9 Traffic flow example with selected flow of about 3 NM (see text)

Both examples are based on an identically high number of approaches per day, with similar distributions of four characteristic peaks.

In the day 1 example (see figure 8), with FLOW selected as 5 plus/minus 1 NM separation all over the day, there was always a jumpy increase of delays at the peaks, which piled up for hours until they could be reduced at less busy times. The horizontal flight paths of the late-morning peak showed an accumulation of holding patterns.

In the day 2 example (see figure 9), FLOW was predominantly set to 3 NM separation. This led to a much higher traffic flow and a clear traffic pattern within the TMA. Peaks were worked off more quickly, and the number of delays was always kept low. In the horizontal flight paths (of the same interval as above) holding patterns are completely missing.

It is not a surprise that low minimum separation is favourable to a high traffic flow. But, in the case of COMPAS, where the exact quantitative variation of this key parameter is handled by controllers in a most direct and simple way, it is important that, via FLOW function, the separation is always kept at the lowest possible level. However, as this is a matter of many factors (e.g. weather), a general demand for lowest FLOW selection, in order to avoid delays or even increase runway capacity, is not admissible.

From this point of view, the handling of the FLOW function by the APPROACH controllers is interesting, as it developed through six months of practice with COMPAS. Figure 10 shows the distribution of the FLOW applied during the first two months and the last months of the evaluation period. Intervals of CAT II/III operation have been canceled in both cases. There was an obvious tendency to select more favourable low-separation values, at the time of advanced experience with the planning system.

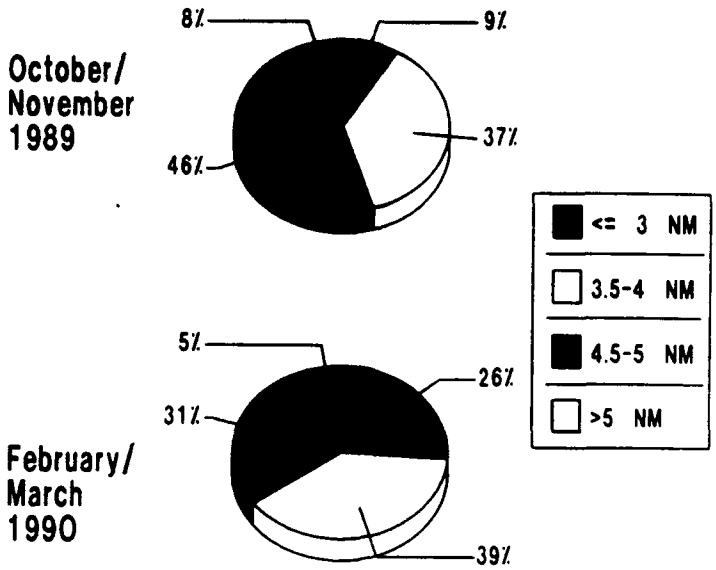
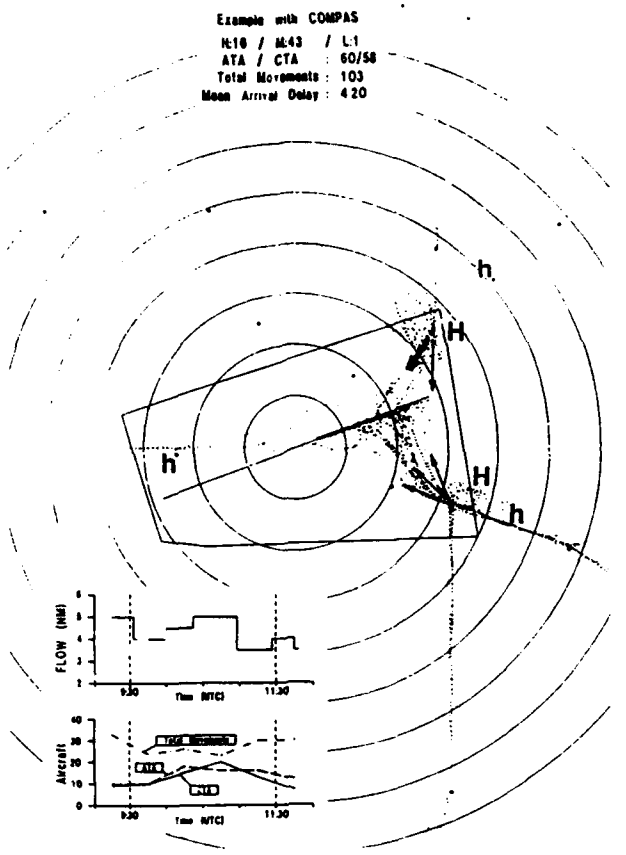


Figure 10 Selected FLOW; percentage of minimum separation classes (in Nautical Miles) selected during the first and the last two months of the evaluation period.

As regards traffic flow in the APPROACH area, the early decisions on final sequencing proposed by COMPAS enabled the controllers to apply, in general, a more direct, straight-in guidance, whereas before more indirect vectoring with flight path stretching was quite usual. Figure 11 shows two typical examples, both referring to two-hour peak traffic periods which were very similar from their basic parameters like landing rate (ATA frequency), mean arrival delay, etc..

APPROACH controllers judged the COMPAS-planned sequences as being easily applicable. Coordination with ACC control was also judged as being significantly easier. Acceptance of the planning system was generally high throughout APPROACH control.



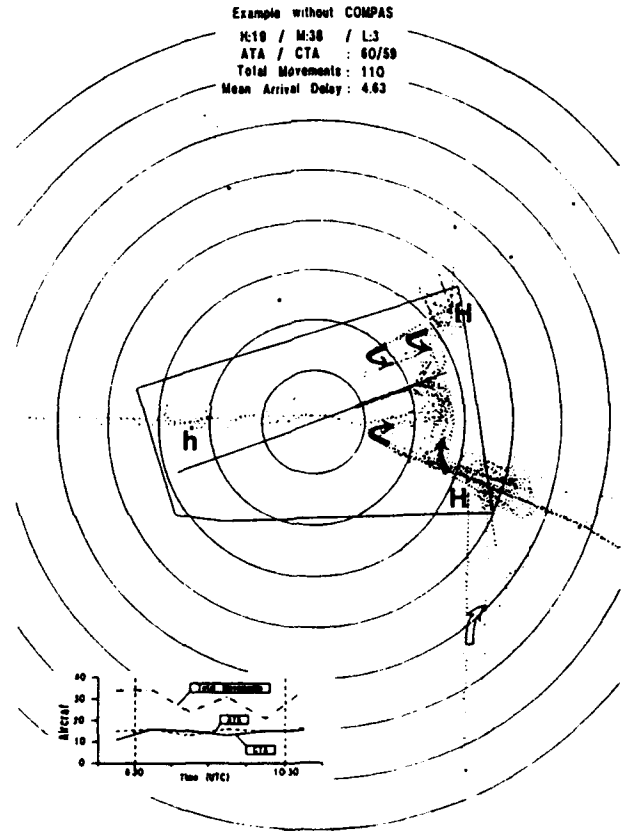


Figure 11 **Examples for horizontal vectoring (top) in the Approach area with COMPAS and without COMPAS (bottom)**

ACC Control

According to the COMPAS philosophy, the individual aircraft's time budget should be balanced before they leave the Metering Fix inbound (TMA boundary), in order to fit smoothly into the overall-planned final approach sequence. So, the effort for doing this is up to the ACC controllers. Although the COMPAS information over all sectors was strongly appreciated by the ACC controllers, putting the COMPAS plans into practice was criticised by many of them as bringing an increase of their control load.

In total, acceptance of the system by ACC controllers was distinctly lower than by APPROACH controllers.

This can at least partly be ascribed to inconvenient locations of the COMPAS displays at some controller working desks, which had to be chosen because of technical reasons.

However, as could be observed from the final application of the questionnaire to this group, there was a significant increase of acceptance regarding the COMPAS planning results. This was obviously due to several program refinements which had been implemented during the evaluation period in order to cope e.g. with some locally specific requirements within the Frankfurt airspace.

Looking at the results of the ACC controller efforts to contribute their part to the overall plan, it has to be stated that metering accuracy (i.e. the average difference between planned and actually observed time overhead the Metering FIX) stabilized very quickly at a constant level. Figure 12 shows the metering accuracies at the two most important fixes for Frankfurt, as calculated for specified time intervals of the test period. It indicates that controllers succeeded very quickly in delivering the aircraft from their sectors always within tolerable time limits to APPROACH control, according to the COMPAS plan.

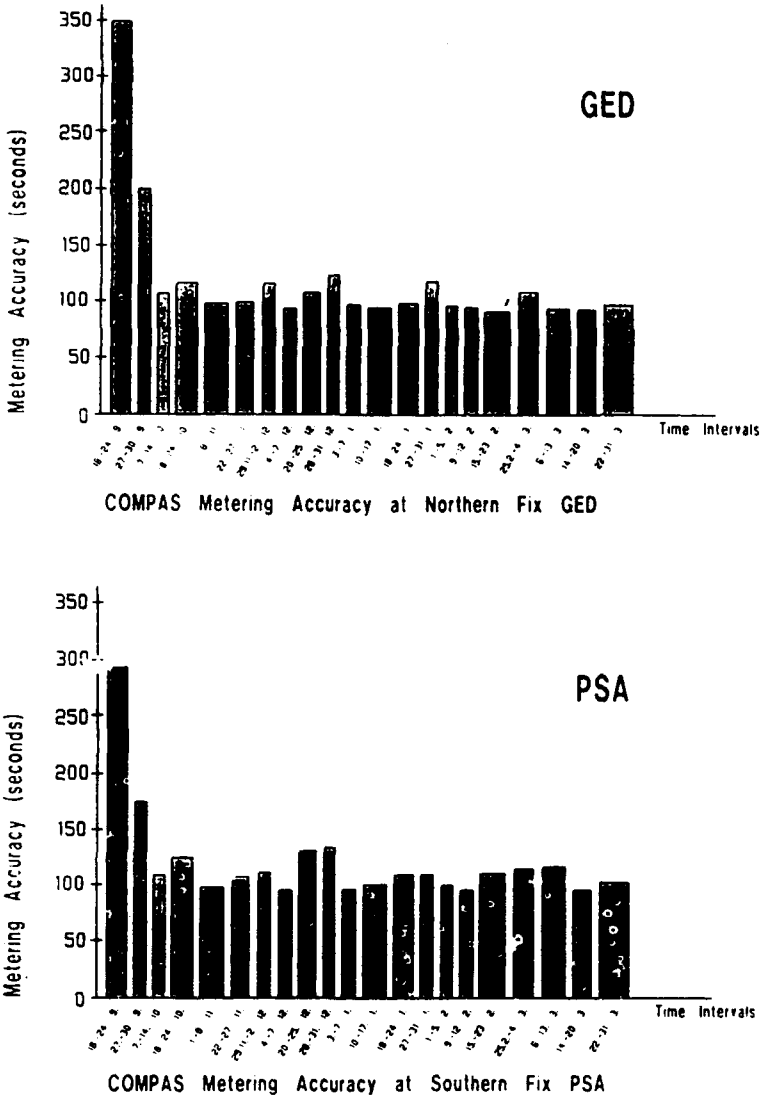


Figure 12 Metering accuracies for specified time intervals of the evaluation period

At the beginning of the test period, this high degree of conformity with the planned traffic flow was achieved at the expense of many manual inputs into the planning system. Figure 13 shows the development of keyboard input frequencies over the whole test period. As can be seen, there was a tendency to decrease towards the end, finally leading to an average of about 15 keyboard entries per hundred approaches.

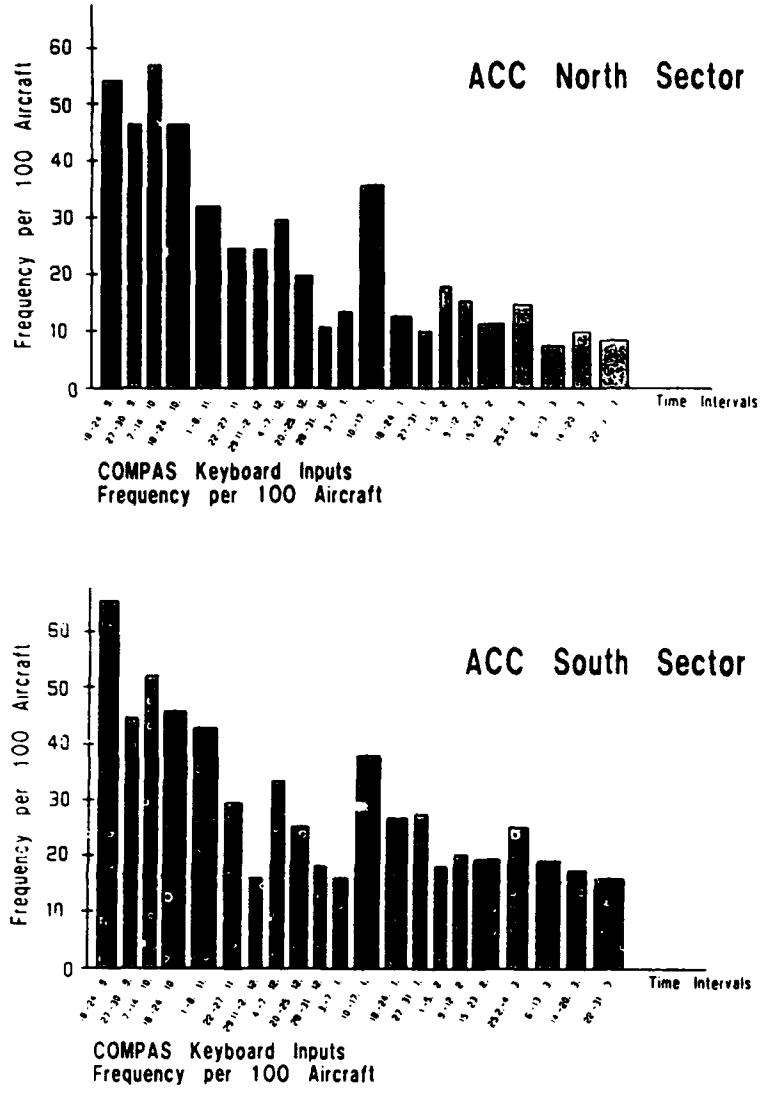


Figure 13 Keyboard input frequencies for specified time intervals of the evaluation period

This tendency presumably reflects several factors, among which at least the following can be named:

- proceeding experience reduced initially prevailing tendencies to try out all system functions,
- program refinements improving planning performance reduced the necessity of making manual inputs, as well as
- an increasing tendency towards low-separation FLOW selection made control of traffic easier and actions, e.g. to reduce an aircraft, less frequent,

which then, taken altogether, might have been the reason for the raising acceptance of COMPAS planning results.

Approach Traffic as a Whole

As a finding which holds for the approach traffic as a whole, observations of better first-come-first-served compliance with the planning system were further supported. Figure 14 contrasts pairwise COMPAS- with non COMPAS-sequences from two-hour traffic peaks which were very similar in all other respects. In general, rank displacements with COMPAS were smaller than before.

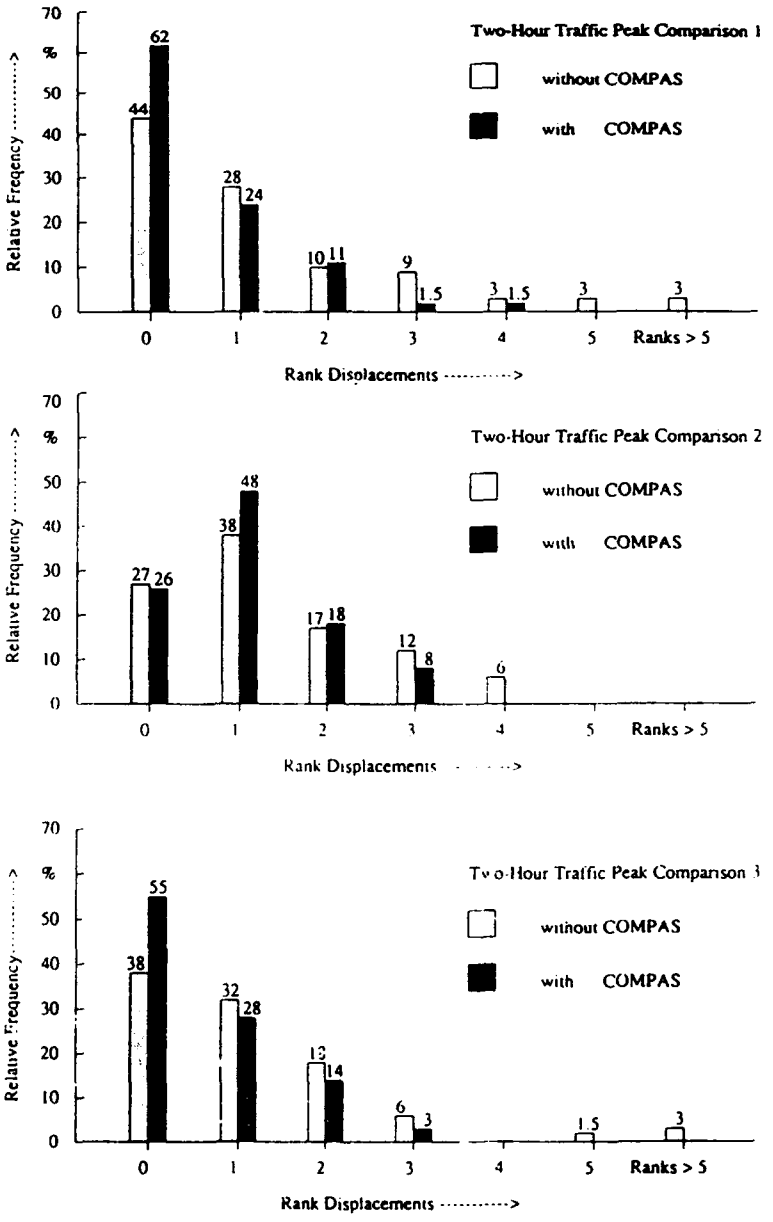


Figure 14 Rank displacements from first-come-first-served sequence

So, there is evidence that making use of the planning aid, thereby keeping the FLOW always at an appropriately low separation, contributes to straight and easy traffic pattern as well as to the fairness of approach sequences.

5. Conclusions

For an evaluation and implementation of an operational version of a planning system in ATC like the COMPAS-System in terms of traffic flow, system handling and controller acceptance, a six-month field test was carried out. The results indicated that the planning system enabled controllers to establish preplanned optimized approach sequences and contributed to straight and easy traffic flows.

As air traffic controllers will continue to play a significant role in the main control loop, the achievement of acceptance is a prime business when to implement planning functions as new ATC system elements.

6. References

[1] Völckers, U. Dynamic Planning and Time Conflict Resolution in Air Traffic Control In: Lecture Notes in Control and Information Sciences, Artifical Intelligence and Man-Machine Systems (1986)

[2] Schubert, M. Modelle für die bodenseitige Anflugplanung COMPAS DLR-Mitteilung 1986-24

[3] Schubert, M. Völckers, U. Schick, V. Ergebnisse der Simulatorerprobung des Planungssystems COMPAS DLR-IB 112-87/02 1987

[4] Schick, V. Völckers, U. The COMPAS-System in ATC Environment DLR-Mitteilung 1991-08

ICAAS PILOTED SIMULATION EVALUATION

by

R.P. Meyer, R.J. Landy and D.J. Halski
McDonnell Aircraft Company
P O Box 516
St Louis, MO 63166
United States

Abstract

This paper describes the Integrated Control and Avionics for Air Superiority (ICAAS) piloted simulation evaluations, along with the system development leading up to the simulations and the data analysis plan for evaluating simulation results. The ICAAS advanced development program is sponsored by the United States Air Force Wright Laboratory at Wright Patterson Air Force Base, Ohio. A research and development contract was awarded to the McDonnell Aircraft Company, St. Louis, Missouri, in September 1987. The program objective is to develop, integrate and demonstrate critical technologies which will enable United States Air Force tactical fighter "blue" aircraft to kill and survive when outnumbered as much as four to one by enemy "red" aircraft during air combat engagements. Primary emphasis is placed on beyond visual range (BVR) combat with provisions for effective transition to close-in combat.

This paper provides an overview of the ICAAS system and its functions. The hardware elements needed to implement the ICAAS system are described and require high throughput parallel processing Reduced Instruction Set Computer (RISC) computers and an advanced "glass" cockpit.

The development of the ICAAS system is discussed as it moved from rapid prototyping to unit testing of attack management and flight Management software and finally into hardware software integration testing prior to manned simulation.

The ICAAS program is structured to evaluate system performance in a build up manner from 2 blue vs 2 red scenarios to 4 blue vs 16 red scenarios. Both offensive counter air and defensive missions will be simulated. The simulation configurations for both the 2v8 simulations which are taking place in 1991-1992 and the 4v16 simulations which will take place in 1992-1993 are presented. A matrix of test blocks for the 2v8 simulations is included.

The Data Analysis plan for reducing piloted simulation results to assess system performance is described. The evaluation of the ICAAS system performance encompasses not only integrated system performance but also performance of individual ICAAS subsystems as they are reflected by measures of situation awareness, flight management utilization, weapon utilization and sensor effectiveness. This includes use of multiple measures of performance as well as univariate and multivariate analyses to assess ICAAS system performance. (Specific simulation results are classified and are not included in this paper.)

System Introduction

Specific functions of the ICAAS system include attack management, tactics, attack guidance, defensive assets manager, and aircraft performance monitor as illustrated in Figure 1. The latter four are collectively referred to as flight management.

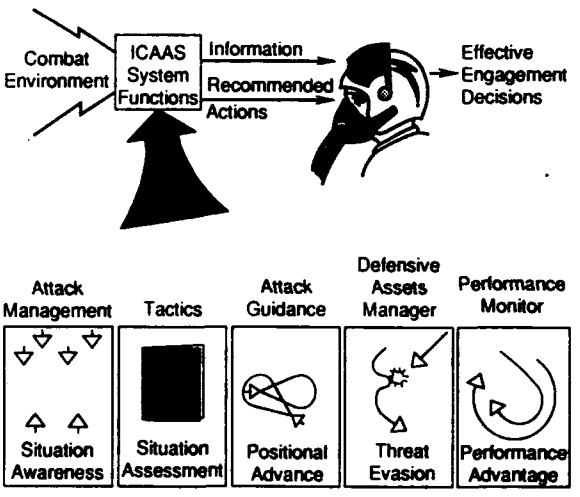


Figure 1. ICAAS System Functions

Attack management automatically controls onboard sensors and computes a correlated summary of track files from ownship sensors as well as information received over the data link from other flight members. The track file summary is displayed to the pilot to maximize his situation awareness.

The tactics function uses a rule-based approach which blends a number of knowledge-based values to assist the pilot in assessing the overall situation. Scenario attributes are evaluated in real-time against a tactics knowledge base, extracted from pilots, to recommend the most viable tactics. Specific coordinated assignments are provided to each flight member.

Attack guidance provides flight trajectory information for achieving missile launch solutions against multiple enemy aircraft while maximizing ownship survival. Faster than real-time aircraft and missile flyout models are used to perform engagement predictions and guide the aircraft to recommended launch points.

The defensive assets manager is activated by the sensed launch of a threat missile against ownship. An extensive data base of early, midcourse, and endgame maneuvers is used to generate and validate available evasive options and to select the most promising for recommendation to the pilot.

The performance monitor uses stored knowledge of the aircraft performance characteristics to aid the pilot in achieving the best real-time dynamic aircraft performance. The features of these specific ICAAS system functions are addressed in more detail in Reference (1).

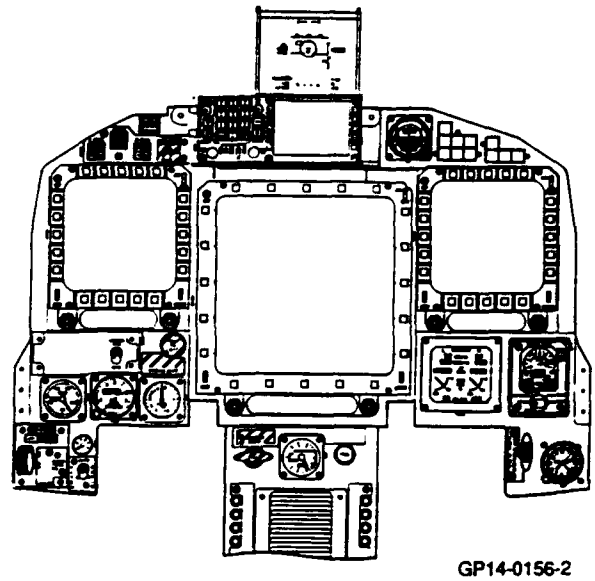
System Implementation

The Department of Defense standard Ada software language is being used for ICAAS implementation. Most of the software code has been developed, including 100,000 lines or over 850,000 32-bit words of executable code plus 275,000 words of data base. Rapid update rates which are necessary to support real-time mission performance led to a computer throughput requirement of approximately 15 million instructions per second. No available flightworthy computer with this capability could be found, so a new computer was developed based on the MIPS Corporation R-3000 32-bit reduced instruction set computer (RISC) processor. This is one of two standard processors selected by the Joint Integrated Avionics Working Group. Each processor is conservatively rated at five million instructions per second. Three processors are included in a single computer chassis, resulting in a throughput capacity of 15 million instructions per second. One chassis hosts the tactics, attack guidance, defensive assets manager, and performance monitor functions and is designated the ICAAS Integration Computer (IIC). The second computer, the ICAAS Support Computer (ISC), hosts the attack management and other ICAAS functions. These computers are flightworthy and will be used in simulation evaluations as well as flight test.

Pilot/vehicle interface (PVI) is a major ICAAS system integration issue. No matter how much information can be developed within the system, effective combat performance can be achieved only if the pilot can easily understand and apply the information. Too much data and information can easily overwhelm the pilot. Efficient cockpit data management techniques are needed, including a proper balance between automation and manual task allocation.

An advanced cockpit instrument panel, designated Cockpit 2000, has been designed for ICAAS as shown in Figure 2. Three multifunction color displays are used to present tactical situation data, weapon delivery information, threat warnings, recommended tactics, and flight trajectories. A large center display provides 9-1/2 inches by 9-1/2 inches of viewing area as the primary situation awareness display. The other two displays are 6 inches by 6 inches. A helmet mounted display (HMD) is provided for sensor cueing and threat warning. The HMD will also be used to cue the pilot on where to look for targets as he transitions from BVR to WVR, thus providing maximum visual acquisition range.

Inflight data entry will be managed through the Up Front Control (UFC) panel which is located just below the HUD. Weapon selection, target designation, and display management functions are provided through standard hands-on throttle and stick switches, or by switches located around the perimeter of the three multifunction displays. An additional method is provided by a touch sensitive overlay for the large central display.



GP14-0156-2

Figure 2. Cockpit Instrument Panel

Pilot working group members have been involved in a series of static mockup, rapid prototyping, and part task simulation experiments as the specific cockpit layout and display formats have evolved. It is essential to achieve a pilot-friendly implementation of the ICAAS technology.

System Development

The ICAAS system design has been developed in a process that began with rapid prototyping on workstations, continued with detail design and verification of the major system elements of attack management and flight management, and culminated with hardware/software integration and testing in MCAIR's Advanced Integrated Control (AIC) laboratory. This design is being further refined and evaluated in piloted simulation and will be verified in limited flight tests.

At the heart of ICAAS development test and evaluation is the Air Combat Engagement System (ACES) which has been used at every stage of the development process and is now described in detail.

Air Combat Engagement System - ACES is a computer based emulation of tactical air combat. The ACES design and implementation in the ICAAS system is based on extensive MCAIR experience in developing and employing a testing and training capability that is embedded on-board the fighter aircraft. This unique capability, referred to as on-board simulation (OBS), was first developed and demonstrated in the Integrated Flight Fire Control (IFFC) Program for bombing and air-to-air gunnery weapon modes. IFFC program engineers used the mode extensively for system checkout, refinement, and integration testing in the lab, in the manned simulator and on the aircraft on the ground. Test pilots flew the OBS mode to gather flight test data on the performance of the IFFC system. ACES was developed by MCAIR in a series of Wright Laboratory sponsored programs known as Onboard Simulation - Enhanced to handle WVR and BVR missile engagements.

The ACES implementation approach for ICAAS is illustrated in Figure 3. When the pilot selects ACES, the digital aircraft capabilities and flight paths, and the relative geometry between the ownship and digital aircraft, are determined by software resident in a computer on the host aircraft. The ACES software is integrated with the ICAAS system such that the cockpit controls and displays respond in the same way as they do in tactical situations against real targets. As a result, the fighter aircraft operations when using the ACES mode are similar in all important aspects to those used in tactical situations. Moreover, if the ACES mode is not selected, it does not alter the tactical operation of the fighter aircraft.

The current ACES capabilities in ICAAS include: (1) eight digital aircraft with a further increase to sixteen planned for the 4v16 simulations, (2) threat tactics consist of current intelligence information, (3) internetted cooperative blue aircraft operation, (4) missile scoring for the AMRAAM (AIM-120) and two threat missiles, the AA-10C and AIM-9L (emulating a threat IR missile), and (5) countermeasures (electronic and mechanical) effects on the blue and red missiles.

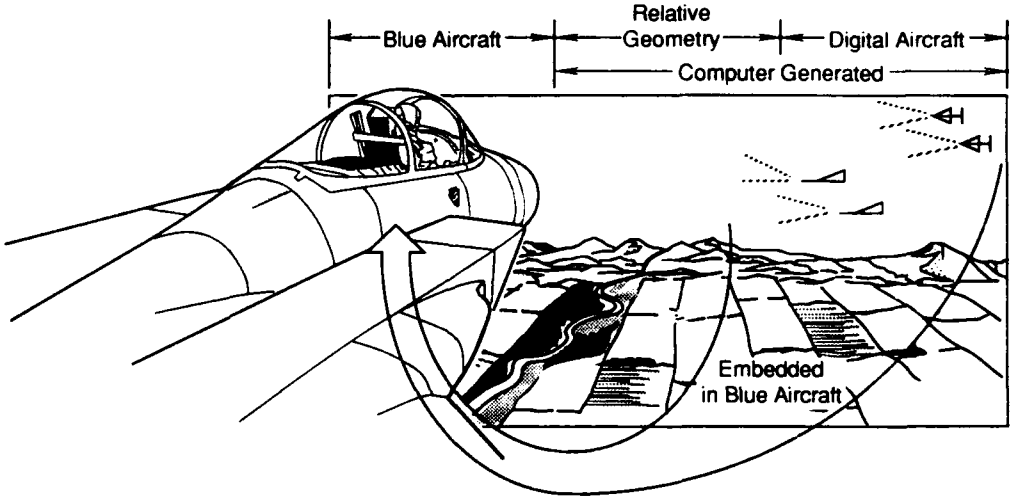
In the ICAAS program, ACES will enable the same engagements to be flown in the manned simulator, in the software test facility, at the aircraft during ground check out, and in flight. ACES should significantly reduce the need for special ICAAS ground support equipment to support flight test while providing a dynamic closed loop testing and checkout capability. Without the ACES mode, it would be difficult for ground support equipment to provide this capability.

Another very important capability provided by ACES is the relatively large number of engagements that can be flown in a flight hour when compared to the number of engagements flown when using tactical aircraft for targets. These engagements will be used for checkout, development, data recording and pilot training on the ICAAS system in flight.

ACES has the capability to be used in the ICAAS flight test with two blue aircraft. Threat truth data generated by ACES on the lead aircraft will be sent over the intra-flight data link to the wingman. Then the wingman will execute ACES sensor models to accept threats that would be seen by his own sensors. The missile scoring algorithm in each blue aircraft will be called to score ownship launches and threat launches on ownship.

Rapid Prototyping - Rapid Prototyping (RP) workstations have been in use since early in the program and continue to provide the MCAIR ICAAS program with a rapid prototyping capability for real time evaluation of the ICAAS system. These workstations have been used by flight control, fire control, and displays engineers to obtain much better insight into the dynamic operation of ICAAS candidate designs than could be obtained using batch analysis programs. This type of insight is not normally obtained in most programs until the manned simulations phase, which is very late for making system design changes.

The RP workstations were used in the program by software engineers to get a preliminary indication of the computer throughput and memory requirements for ICAAS software. As ICAAS software development continued, the workstations have been used for debug, checkout, software interface definition, and unit testing. The RP workstations have been expanded to include 1553 links to additional processors. These 1553 links allow the ICAAS computers housing ICAAS software to be included and used for systems checkout prior to testing



ACES
- Emulates Tactical Air Combat
- Integrated With Aircraft Systems
- Does Not Affect Aircraft Tactical Operation

ACES Features
- Multiple Threats Cooperative Threats
- Blue/Neutral/Red Digital Aircraft Maximum of 8
- EW Effects
- BVR to WVR Engagements
- Missile and Gun Scoring, ECM Effects
- Supports 2vN Engagements

GP14-0156-3-D/kch

Figure 3. Implementation Approach and Features of the Air Combat Engagement System (ACES)

with the rest of the ICAAS system computers in the integration laboratory and the manned simulator.

Unit Testing - A specific application for the rapid prototyping facility is its use in the integration, verification and validation of the operational flight programs for the ISC computer. Figure 4 illustrates the MCAIR RP environment which has been set up for verification and validation of the ACES, AMIC and emulated ICAAS Attack Management (IAM) code in the ISC. This was used prior to the availability of the actual IAM code.

The IAM associate contractor, Northrop, is using the RP environment shown in Figure 5 for the integration, verification and validation of the IAM operational flight program in the ISC computer. Lead/wingman aircraft operation is being simulated to check out the internetting function of the IAM software.

MCAIR's subcontractor, Lear Astronics, has also acquired an RP environment shown in Figure 6 which they are using for the integration, verification and validation of the Flight Management System (Tactics, Attack Guidance, DAM, APM) in the IIC. Using the configuration shown, both lead and wingman Flight Management System operation have been checked out.

AIC Laboratory Integration Testing - The laboratory configuration (Figure 7) was developed to study two aircraft IIC/ISC interface issues. This configuration uses the RP workstations to simulate the Central Computer and data link. A IIC/ISC pair of computers is linked to each workstation.

The configuration shown in Figure 8 has been used in the period preceding 2vN Manned Simulation testing. It simulates a second F-15 aircraft equipped with all ICAAS hardware. A second set of ICAAS hardware (CC, IIC, ISC) was installed on a second (F-15) test bench. Information is passed by a data link simulator. It provides an independently controllable wingman which is using the same OFP as the one used in manned simulation and flight test.

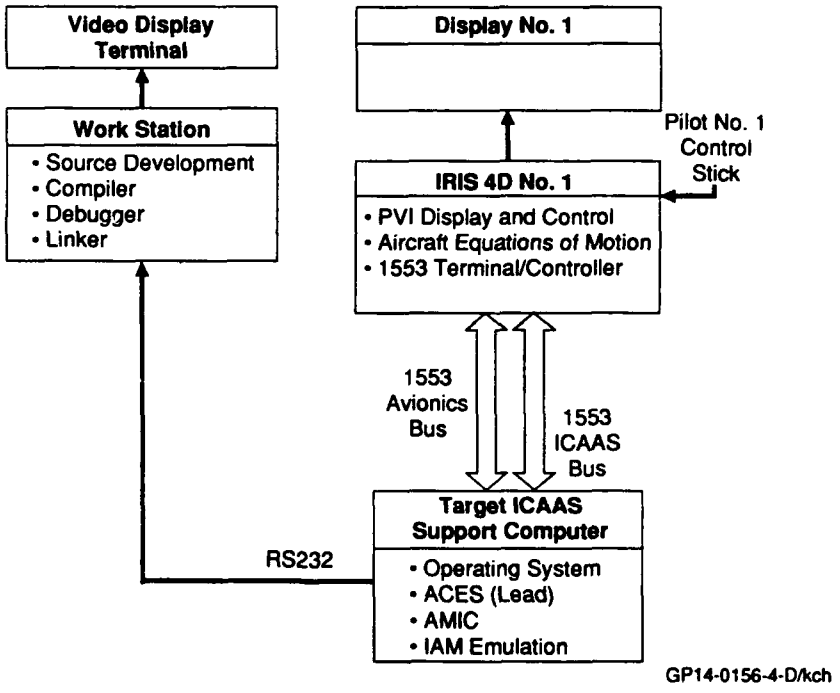


Figure 4. MCAIR ACES/AMIC/ISC Rapid Prototyping Configuration

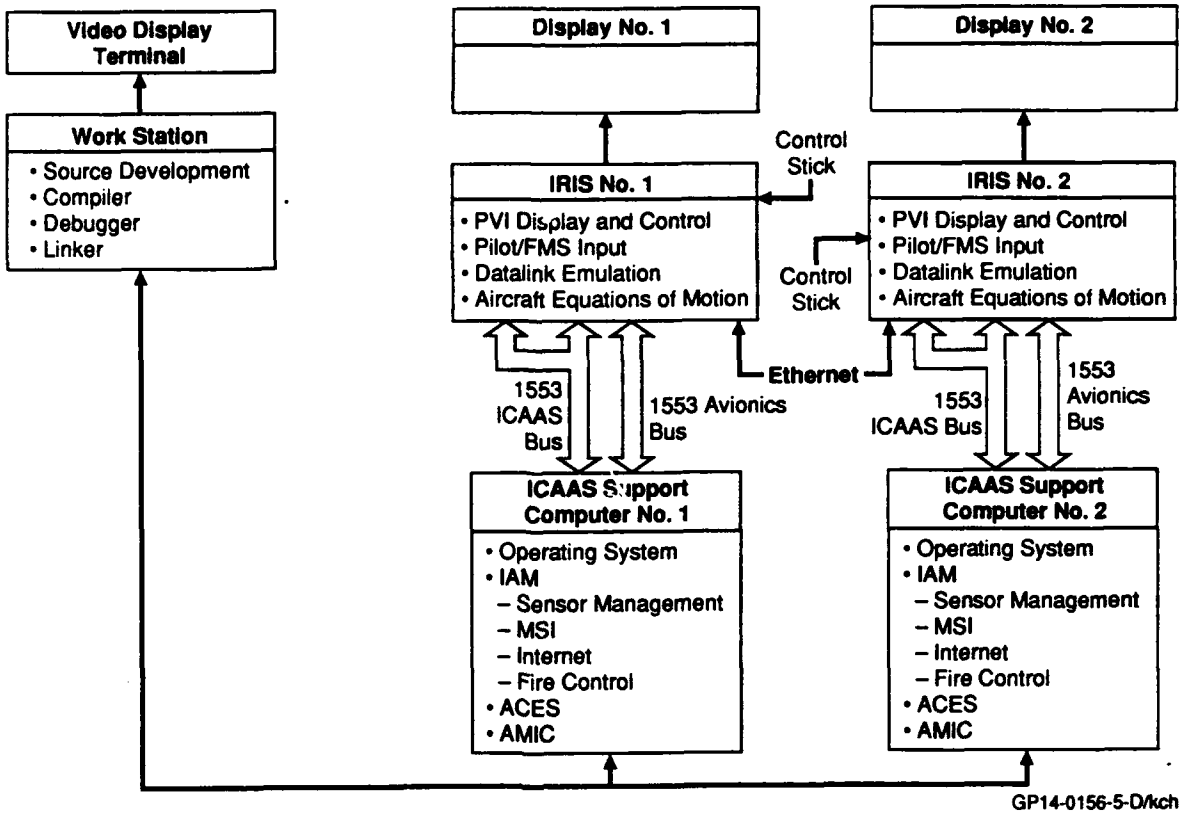


Figure 5. Northrop IAM/ISC Rapid Prototyping Configuration

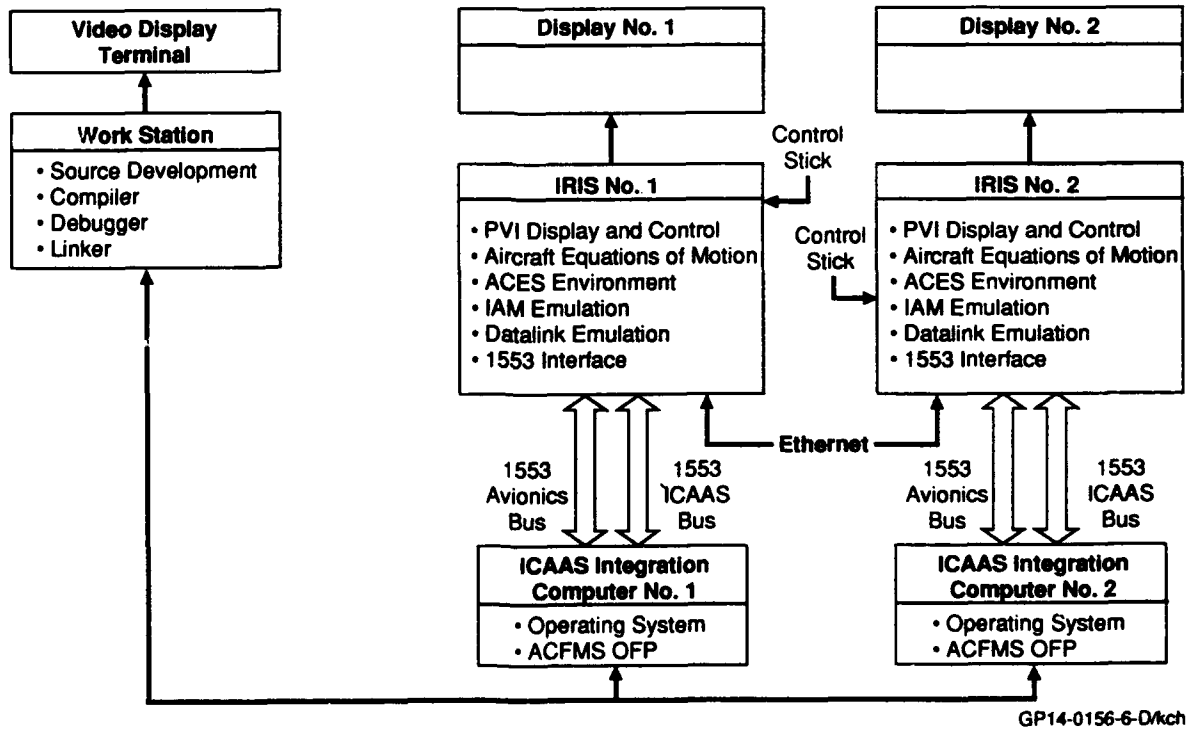


Figure 6. Lear Astronics ACFMS/IIC Rapid Prototyping Configuration

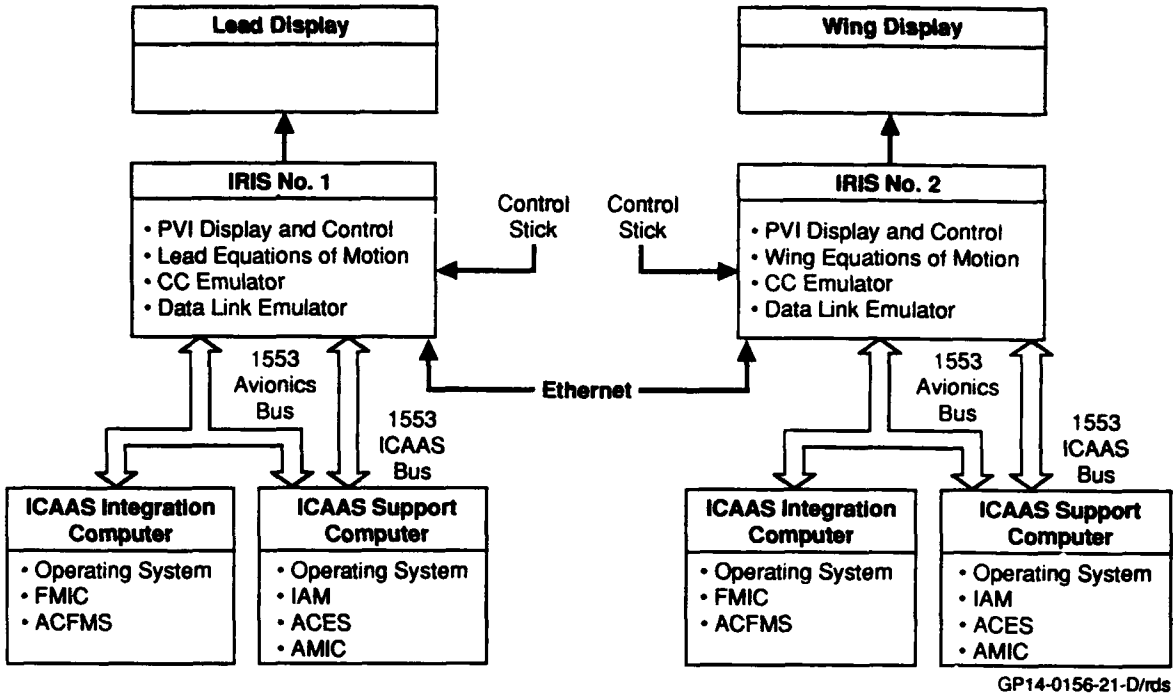


Figure 7. Two Aircraft IIC/ISC Test Configuration

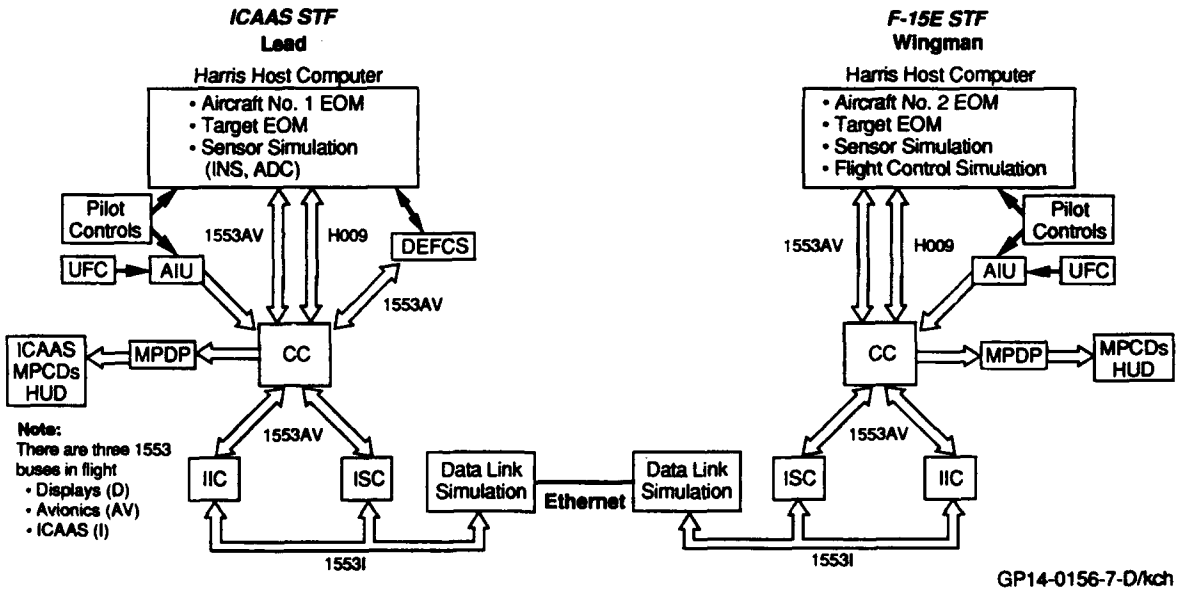


Figure 8. ICAAS/F-15 Hardware/Software Test Facility Configuration, 2vN

Simulation Description

Hardware Configuration - Figure 9 shows details of the ICAAS manned simulation configuration for the 2vN full mission simulations. N represents the number of threats and can vary from 2 to 8. Both blue cockpits have Cockpit 2000, one implemented in a Manned Air Combat Simulator dome, the other in a reconfigurable crewstation. Two sets of ICAAS computational equipment (CC, ISC, IIC) are required. The displays are simulated and controlled by IRIS and IMI computers. Three SEL host computers are required. The Data Link is emulated through the Data Distribution System.

The third and fourth blue aircraft required for the 4v16 simulations will use Super Manned Interactive Crewstations. Each will have the Cockpit 2000 display on a large Cathode Ray Tube and use out the window and HUD displays projected on a screen forward of the crewstation.

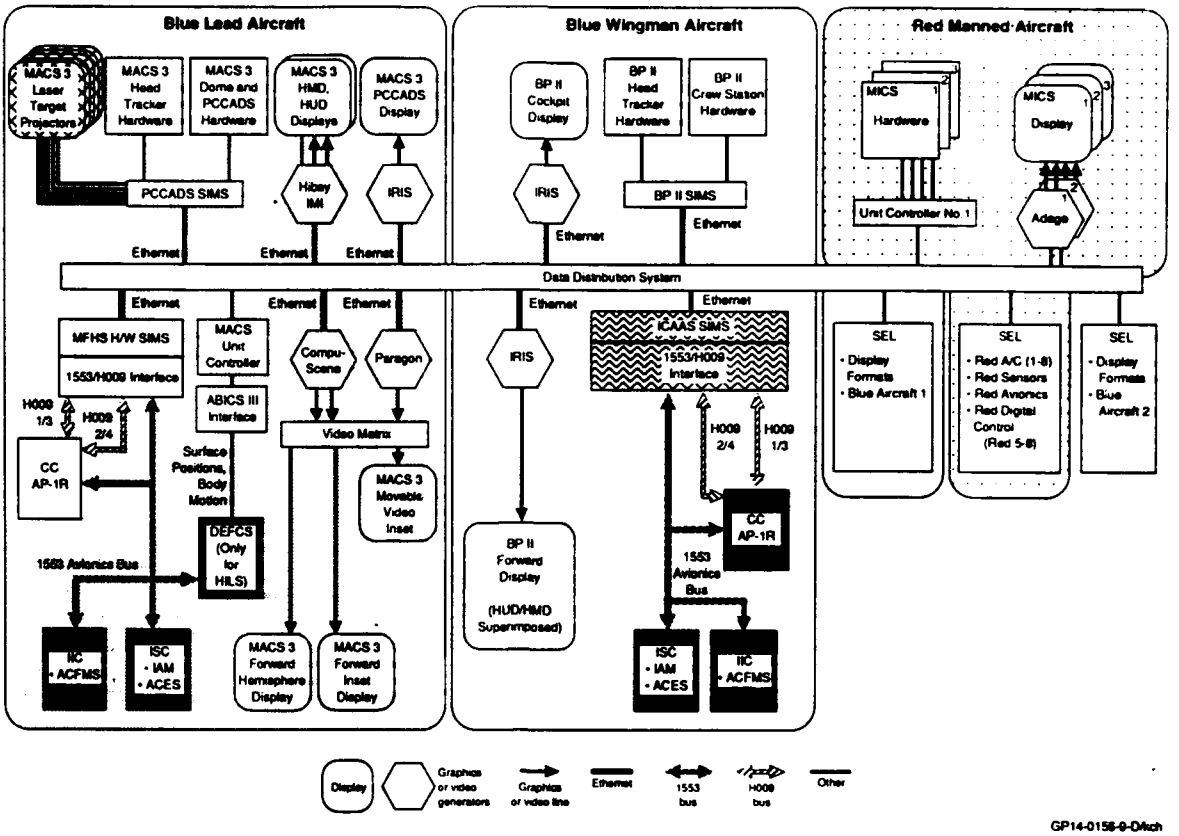


Figure 9. 2VN ICAAS Simulation Hardware Configuration
Full Mission Simulation: 2 Manned vs Up to 3 Manned + 5 Digital

Four sets of IIC/ISC equipment will be required. These will be laboratory nonflightworthy computers different from those used in the earlier 2vN simulations and flight test. We plan to use IRIS workstations to emulate the Central Computer and datalink as shown in Figure 10. Five SEL computers will be required to host the environment of 4 blue aircraft and 16 red aircraft.

Software Configuration - The ICAAS simulations involve a standard F-15 to be used to generate baseline results and an ICAAS equipped F-15. In addition it will also include an advanced aircraft with and without the ICAAS system.

For the baseline results, all simulation software is contained in the simulation SEL computers as shown in Figure 11.

For the ICAAS equipped aircraft simulations, the software is distributed as shown in Figure 12. All ICAAS system software is distributed among the IIC, ISC and CC. The ISC also contains sensor models for the radar, Infrared Search and Track Sensor (IRSTS), Interrogate Friend or Foe (IFF), Electronic Warfare Warning System (EWWS) and Missile Warning Sensor (MWS). This was done since the ISC has more throughput than the SEL.

Scenarios - The ICAAS system will be evaluated against three type of scenarios:

- o AWACS Defense Mission 2v2(F)
- o Base Defense Mission 2v4(F) + 4(B)
- o Fighter Escort Mission 2v6(F)

where F means fighter, and B means bomber. There will be variations in red aircraft initial positions for each of these three missions.

Simulation Test Matrix - The top-level simulation test matrix is presented in Figure 13 and discussed in Reference 2. A necessity in any attempt to quantify performance improvements is to have a well-defined baseline for comparison. The ICAAS test matrix has two such baselines. Block 1 is the F-15 baseline with a Mechanically Scanned Antenna (MSA) radar, no intraflight data link, a standard F-15C cockpit, no Low Observability (LO) features, and no ICAAS flight management or attack management capability.

The second baseline is defined in Block 8. This baseline is representative of the next generation high performance front-line USAF fighter having an Electronically Scanned Antenna (ESA) radar, an intraflight data link, three glass cockpit displays, LO signatures, and an attack management system based on the Air-to-Air Attack Management program. Missing from Block 8 are the ICAAS flight management functions, whose performance increments will be obtained by comparing Block 9 results with the Block 8 baseline. The performance increments resulting from applying full-up ICAAS functions to a current day fighter will fall out of the comparison of Block 3 to Block 1 results.

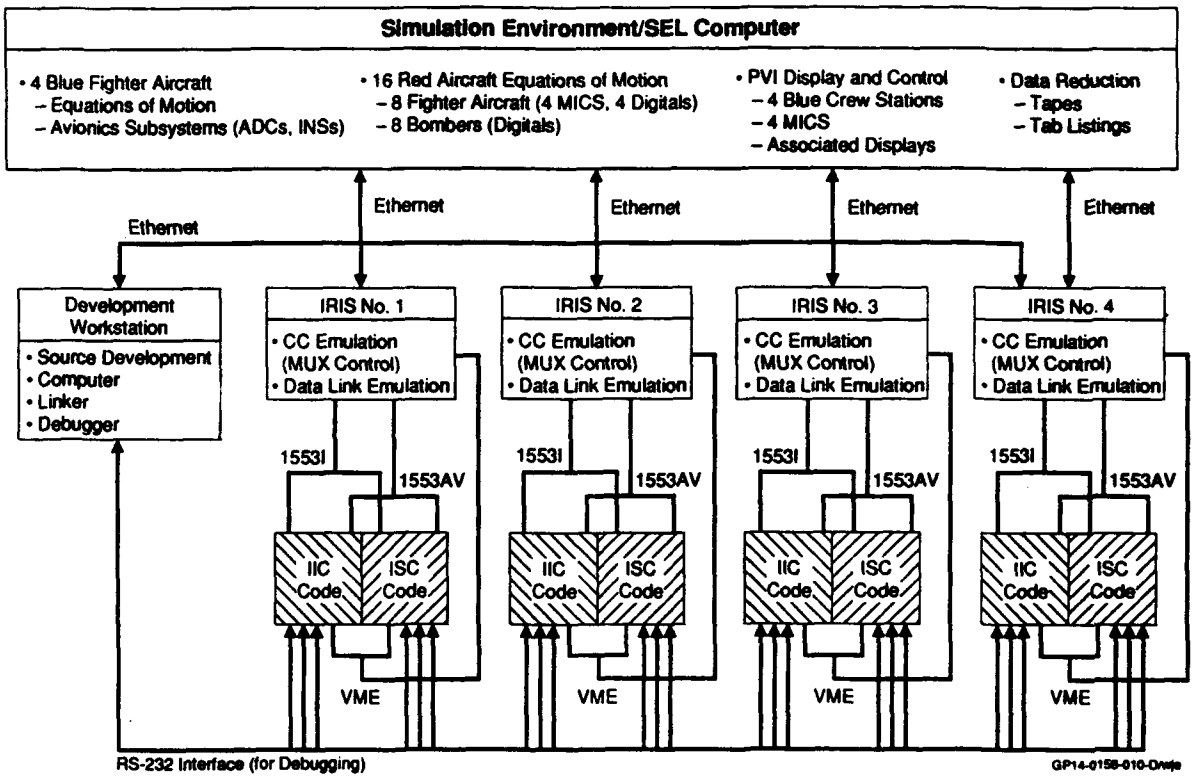


Figure 10. 4 v 16 ICAAS Simulation Configuration

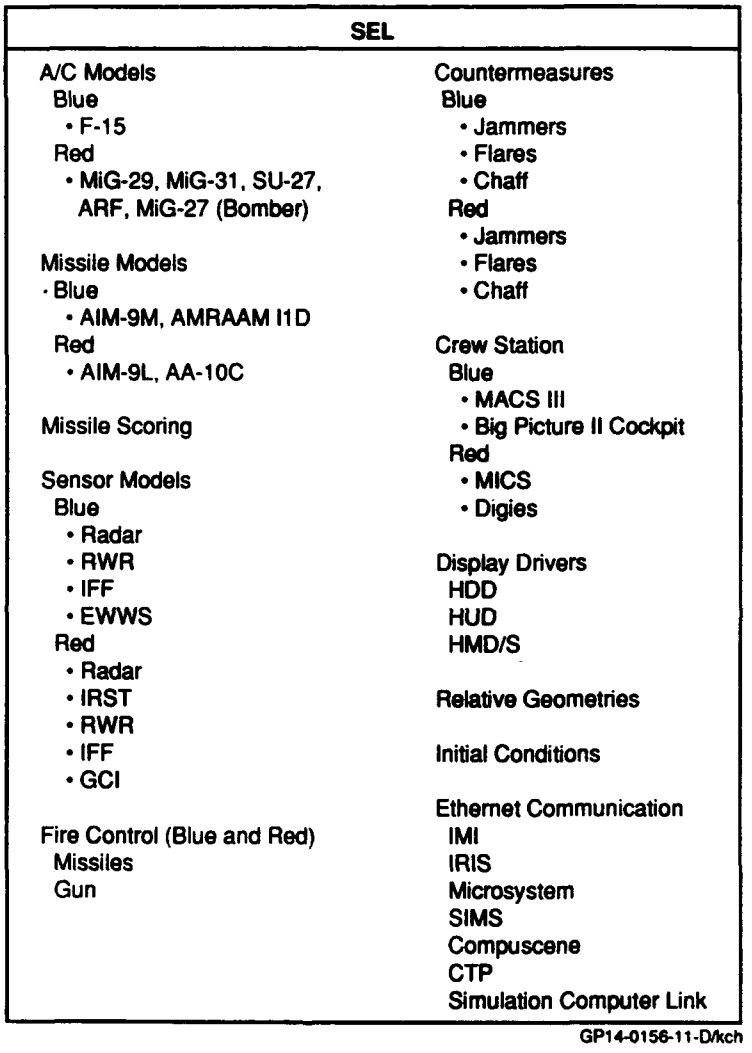


Figure 11. Baseline F-15 Software Configuration

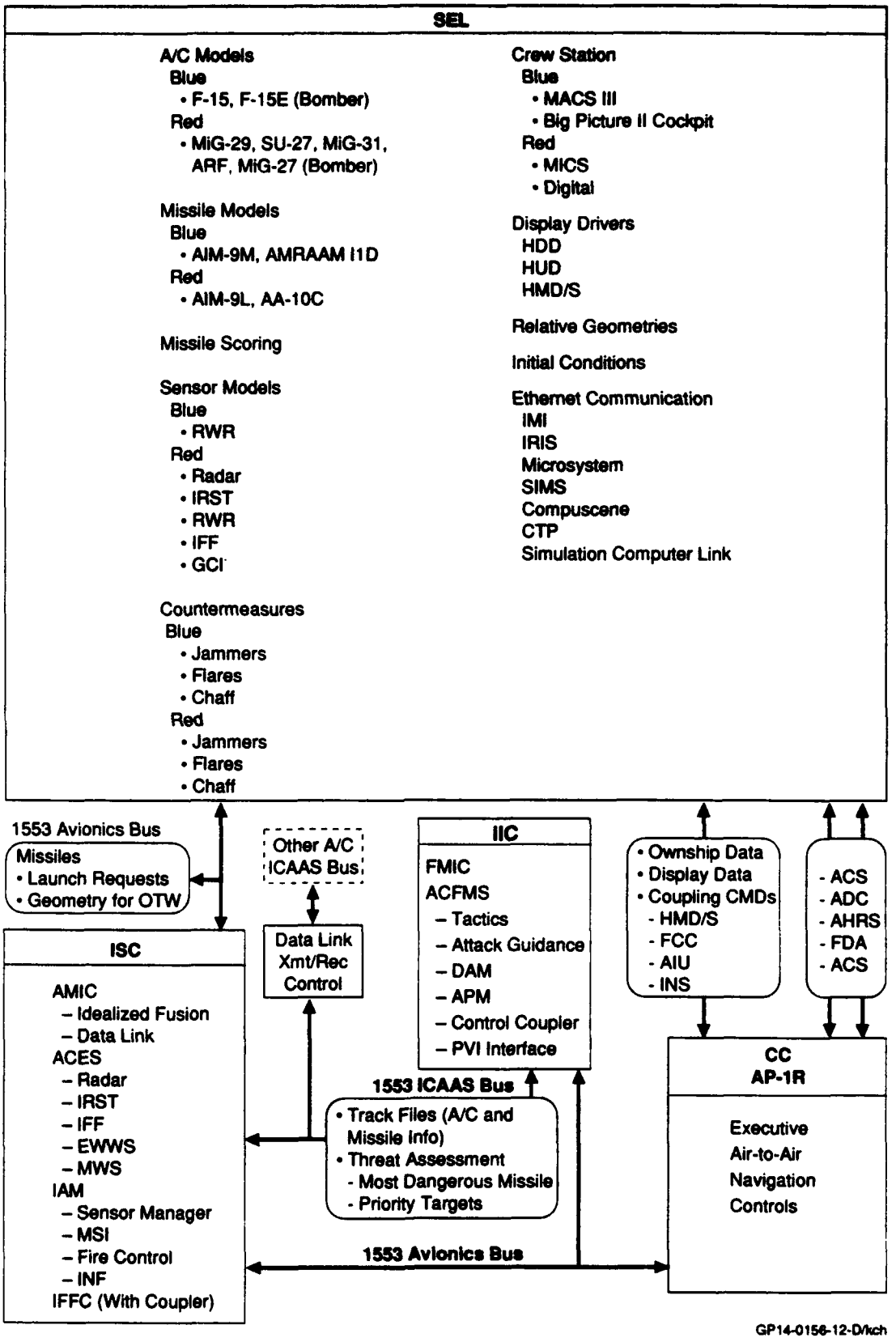


Figure 12. ICAAS Simulation Software Configuration

It can be seen by examining the Figure 13 test matrix that other blocks were designed to also quantify the individual performance increments resulting from an ESA radar, an attack management system, LO technologies, and an intraflight data link. Other blocks will investigate the effects of weather (Block 12) and of the Electronic Warfare (EW) environment (Blocks 6 and 13) on ICAAS system performance. Block 7 will produce manned simulation results that can be compared directly to data obtained from flight tests.

Three separate and major manned simulation sessions are planned at the contractor facilities under the ICAAS program. Each will use the basic test matrix, but system capabilities will be varied to correspond to performance levels typical of the years 1995, 1998, and 2002. Countermeasure technology progresses from current day chaff and flares to conceptual RF towed decoys in the last simulation session. Self protection system ranges and angular coverages increase between simulation entries. Data link ranges and sophistication of the tactics algorithm also progress. AMRAAM improvements are postulated which would increase its range and radar sensitivity in the last simulation entry. Corresponding performance improvements are also granted to enemy capabilities. For instance, while the AA-10C is the threat medium range missile in the first simulation, the projected Soviet FOMRAAM (Follow On Medium Range Air-to-Air Missile) becomes the primary threat missile for the last of the three simulation sessions.

Data Analysis

The ICAAS evaluation simulation will employ subjective and objective measurement to assess ICAAS system success. These measurements are: workload, situation awareness, pilot acceptance, and objective data. Video tape of cockpit displays will also be recorded and used to assess the operation of the ICAAS system.

Workload - The Subjective Workload Assessment Technique (SWAT), developed by the Air Force Armstrong Aerospace Medical Research Laboratory (AFAAMRL) will be used for workload measurement. This measure is relatively unobtrusive and requires little effort from the participant.

SWAT treats workload as comprised of three components. These components are time required by the task, effort required by the task and the stress imposed by the task. Participants rate each component individually on a scale from one to three, as shown in Figure 14. These rankings are analyzed mathematically to produce a single number which rates total workload on a scale from zero to one hundred, zero being the minimum workload, one hundred being the maximum.

Situation Awareness - The situation awareness rating scale to be used are shown in Figure 15. Situation awareness items will be combined with workload and pilot acceptance items and administered in written form to each pilot after each run.

Pilot Acceptance - The ICAAS system offers innovative solutions to pilot situation awareness and situation assessment problems. Regardless of the intelligence built into these systems, they cannot assist pilots unless pilots accept them and can use them as useful aids in the conduct of their mission. Questionnaires administered after each run, at the completion of each test block, and at the end of the test period will assess pilot opinion on the concepts and implementation of the ICAAS systems including the Tactics Algorithm, Attack Guidance, data link, IAM and the Cockpit 2000.

Blk	Friendly/Threat Detectability	Aero & Perf	PVI	Sensors/ Att Mgmt	Data Link	Flight Mgmt	Comments
1	F-15/F-15	F-15	F-15	MSA	Off	Non-ICAAS	Baseline F-15
2	F-15/F-15	F-15	ADV	ESA + IAM	On	Non-ICAAS	Effect of IAM on F-15
3	F-15/F-15	F-15	ADV	ESA + IAM	On	ICAAS	Full-up ICAAS F-15
4	F-15/F-15	F-15	ADV	MSA + IAM	On	ICAAS	Effect of ESA
5	F-15/F-15	ADV	ADV	ESA + IAM	On	ICAAS	Effect of Aero/Perf & ESA
6	LO/Vary	ADV	ADV	ESA + IAM	On	ICAAS	Effect of EW Level 1
7	F-15/EAFB A/C	F-15	ADV	MSA + IAM	Off	ICAAS	1 vs. n Flight Test Config
8	LO/F-15	ADV	ADV	ESA + IAM	On	Non-ICAAS	Effect of Big LO Advantage
9	LO/Vary	ADV	ADV	ESA + IAM	On	ICAAS	Nominal Sim Configuration
10	LO/Vary	ADV	ADV	ESA + IAM	Off	ICAAS	Delta Due to Data Link
11	LO/Vary	ADV	ADV	ESA + IAM	On	Non-ICAAS	Effect of Flight Mgmt
12	LO/Vary	ADV	ADV	ESA + IAM	On	ICAAS	Effect of Weather
13	LO/Vary	ADV	ADV	ESA + IAM	On	ICAAS	Effect of EW Level 2

LO: Low Observable
EAFB: Edwards AFB
A/C: Aircraft
ADV: Advanced
MSA: Mechanically Scanned Antenna Radar
ESA: Electronically Scanned Antenna Radar
IAM: ICAAS Attack Management Function
EW: Electronic Warfare

GP14-0156-13-D/wje

Figure 13. Mission Simulation Test Matrix

I. Time Load

1. Often Have Spare Time. Interruptions or Overlap Among Activities Occur Infrequently or Not at All.
2. Occasionally Have Spare Time. Interruptions or Overlap Among Activities Occur Frequently.
3. Almost Never Have Spare Time. Interruptions or Overlap Among Activities Are Very Frequent or Occur All the Time.

II. Mental Effort Load

1. Very Little Conscious Mental Effort or Concentration Required. Activity Is Almost Automatic. Requiring Little or No Attention.
2. Moderate Conscious Mental Effort or Concentration Required. Complexity of Activity Is Moderately High Due to Uncertainty, Unpredictability, or Unfamiliarity. Considerable Attention Is Required.
3. Extensive Mental Effort and Concentration Are Necessary. Very Complex Activity Requiring Total Attention.

III. Psychological Stress Load

1. Little Confusion, Risk, Frustration, or Anxiety and Can Easily Be Accommodated.
2. Moderate Stress Due to Confusion, Frustration, or Anxiety. Noticeably Adds to Workload. Significant Compensation Is Required to Maintain Adequate Performance.
3. High to Very Intense Stress Due to Confusion, Frustration, or Anxiety. High to Extreme Determination and Self-Control Required.

GP14-0156-014-D/kch

Figure 14. SWAT Rating Scales

	Extremely Good	Good	Fairly Good	Fairly Poor	Poor	Extremely Poor
1. My Knowledge of Friendly Aircraft Position Was	1	2	3	4	5	6
My Knowledge of What Friendly Aircraft Were Doing Was	1	2	3	4	5	6
2. My Knowledge of Hostile Aircraft Positions Was . . .	1	2	3	4	5	6
My Knowledge of What Hostile Aircraft Were Doing Was	1	2	3	4	5	6
3. My Knowledge of My Offensive Options Was . . .	1	2	3	4	5	6
4. My Knowledge of My Defensive Options Was . . .	1	2	3	4	5	6

GP14-0156-015-D/kch

Figure 15. Situation Awareness Rating Scale

Objective Data - An end-of-run summary (Figure 16) and event ledger will be the data used to compute the objective Measures of Performance (MOPs) shown in Figure 17.

The event ledger will be used to record "snapshots" of significant data at critical points such as tactic selection or weapon employment. The end-of-run summary will record data accumulated during the run.

ICAAS Evaluation Simulation (IES-1B)

Date: DD MMM YY

Time: HH:MM Run Number X Scenario

A/C Loss Data

- Red Kills
  - Number of Blue A/C Killed by MICS
  - Number of Blue A/C Killed by Digitals
  - Number of Blue A/C Killed by Blues
- Blue Kills
  - Number of MICS Killed by Blue A/C
  - Number of Digital Fighters Killed by Blue A/C
  - Number of Bombers Killed by Blue A/C
  - Number of Red A/C Killed by Reds
- Losses
  - Number of Blues Hitting Ground
  - Number of Blues Out of Fuel
  - Number of Reds Hitting Ground
  - Number of Reds Out of Fuel
- Mission Success Data
  - Number of Penetrating Threats
  - Number of Red Bombers on Target (ABD)
  - Number of Blue Bombers on Target (Escort)
  - Number of Red Missiles Targeted on AWACS
  - Number of AWACS Killed
  - Number of Missiles Launched at AWACS
  - Average P_k of Missiles Launched at AWACS
- Targeting Data
  - Number of Reds Tracked by Blues
  - Number of Reds Assigned by Blues
  - Number of Reds Launched on by Blues
  - Number of Reds Launching Missiles
  - Number of Blues Tracked by Reds
  - Number of Blues Assigned by Reds
  - Number of Blues Launched on by Reds
  - Number of Blues Launching Missiles
- AIM-120 Data - (Blue Only) - - B1 - - - - B2 - -
  - Number Launched
  - Number Achieving Kill
  - Average Launch Range - Percent R_{max}
  - Number Unguided Before Autonomous
- AIM-9 Data - (Blue Only)
  - Number Launched
  - Number Achieving Kill
  - Average Launch Range - Percent R_{max}
- Fuel
  - Fuel at Start
  - Fuel at End of Run
- Total Time Within MIZ
- Total Time Within Data Link
- Total Mission Time
- Disengage Cue Following
- Total Time in ICAAS Modes
  - Tactics
  - Attack Guidelines
  - DAM
  - GCAS
  - MCAS

GP14-0158-16-Dmss

Figure 16. End of Run Summary Data

Mission Success
- Number of Penetrating Threats
- Number of Red Bombers on Target
- Number of Blue Bombers on Target
- Percent of AWACS Survival
- Average P_k of Missiles Launched at AWACS

Engagement Success
- Exchange Ratio
- Loss Rate Advantage
- Percent Red Aircraft at Which Blue Fired (By Class)
- Percent Surviving Aircraft That Did Not Fire a Weapon (Blue and Red)
- Blue Kills
- Blue Losses
- Blue Valid Launches
- Red Valid Launches

Situation Awareness
- Number of Tracks Initiated by Blue
- Percent of Red Forces Tracked
- Range, Time of First Red Track File Generated
- Range, Time of First Blue Track File Generated
- Number of Hostile IDs Before Preferred Launch Range
- Number of Friendlies or Neutrals Targeted When Tactic Accepted
- ID Contributors
- Percent of MRMs Double Targeted
- Percent of SRMs Double Targeted
- Percent of Opposing Aircraft Targeted
- Number of Times Each Blue Aircraft Launches Weapon
- Frequency of AIM-120 Becoming Unguided in Flight
- Total Time Within MIZ
- Average AIM-120 Launch Range
- Average Percentage of Time Blue Aircraft Are Intervetted
- Number of Launches (SRM, MRM)

Decisions
- Weapon Launch Range by Type Relative to R_{max}, R_{ne}
- F-Poles
- Number of Blue Weapons Launched Before First Red Launch
- Disengage Cue Following
- Disengage Range From Targets' Centroid
- Reattack Range From Targets' Centroid
- Time and Range When Blue Aircraft Committed
- Fuel Usage
- Percent of Assigned Targets at Which Blue Launched
- Duration of Coupled Flight by Mode
- Number of Blue Launches Without Shoot Cue
- Number of Blue Successful Launches
- Valid Red Launches by Quadrant
- Blue Launch Range by Quadrant
- Average End Game Velocities of Missiles Launched at Blue
- Number of Blue Missiles Launched From Each Quadrant
- Number of Valid Blue Missile Launches in Each Quadrant

Degrees of Covertness
- Number of Detects and Tracks by Adversaries
- Percent of Blue Forces Detected, Targeted

Other
- Success of Ground Collision Monitoring - Minimum Altitude
- Success of Mid-Air Collision Monitoring - Minimum Distance Between Aircraft

Subjective Data
- Workload
- Situation Awareness
- Verbal Communication

GP14-0156-18-D/kas

Figure 17. Measures of Performance

Primary Analysis - The data from the ICAAS manned simulations will be analyzed in two stages. The first set of analyses will focus on ICAAS tactical effectiveness using objective MOPs in the mission effectiveness and engagement success categories and the subjective measure of workload.

The multivariate analysis will be a two-way hierarchical analysis of variance (MANOVA) using Number of Penetrating Threats, Loss Rate Advantage, and Blue Workload as dependent variables. The two independent variables will be Test Block, which generally relates to blue aircraft capabilities, and Scenario, which relates to the number and type of the threat. The multivariate tests will quantify ICAAS system tactical effectiveness as it relates to the objectives cited in the Simulation Test Matrix (Figure 13).

The univariate analysis of tactical effectiveness will be based on the measure of effectiveness (MOE) shown in Figure 18. The MOE focuses on effectiveness issues by combining kills and losses data with mission success data using the weights shown. It will be employed as a complementary approach to the analysis of ICAAS tactical effectiveness. The MOE results will be analyzed using the same "completely within-subjects" model and post hoc tests used for the MANOVA described above.

Mission	MOE
AWACS Defense	–0.6 (LRA/TRA) –0.4 (KBF/TBF)
Bomber Escort	0.1 (KRF/TRF) –[0.2 (KBF/TBF) –0.7 (LBA/TBA)]
Base Defense	[0.1 (KRF/TRF) –0.7 (LRA/TRA)] –0.2 (KBF/TBF)

LRA = Leaked Red Attackers.
Equal to number of penetrating threats in MOP list, base defense only.
TRA = Two in AWACS defense. Four in air base defense.
KBF = Blue losses in MOP list
TBF = Number of manned blue fighters. Two in IES-1B.
KRF = Blue kills in MOP list
TRF = Number of red fighters
LBA = Leaked blue attackers. Equal to number of blue bombers on target in MOP list.
TBA = Total blue aircraft

GP14-0156-20-D/kas

Figure 18. Measure of Effectiveness

Subsidiary Analyses - These will focus on issues of special interest in the evaluation of ICAAS systems. These include:

- Situation Awareness
- Flight Management Utilization
- Weapon Utilization
- Sensor Effectiveness

A multivariate analysis will be performed using the hierarchical model and a small number of dependent measures. The dependent measures will be chosen from those objective MOPs relating to Situation Awareness and the subjective ratings of Situation Awareness. Dependent measures will be chosen for their low correlation with other MOPs and their validity as measures of situation awareness.

We will separately assess the extent of ICAAS flight management system usage with the following specific MOPs.

- Duration of ICAAS Modes
- Average End Game Velocities of Missiles Launched at Blue
- Success of Ground Collision Monitoring - Minimum Altitude
- Success of Mid-Air Collision Monitoring - Minimum Distance Between Aircraft
- Disengage Maneuver

In each case the average values and the number of events on which the average is based will be presented.

Weapon use in the test blocks will be compared using the "completely within-subjects" hierarchical MANOVA model. The dependent measures will be MOPs chosen from the following list that have low correlations with other MOPs on the list and have validity as measures of weapon use.

- Blue valid launches
- Percent of MRMs Double Targeted
- Percent of SRMs Double Targeted
- Percent of Opposing Aircraft Targeted
- Frequency of AIM-120 Becoming Unguided in Flight
- Average AIM-120 Launch Range
- Number of Launches (SRM, MRM)
- Number of Blue Weapons Launched Before First Red Launch
- Number of Blue Successful Launches
- Number of Blue Missiles Launched From Each Quadrant
- Number of Valid Blue Missile Launches in Each Quadrant

Sensor effectiveness in the test blocks will be compared using the completely within-subjects hierarchical MANOVA model. The dependent measures will be objective MOPs chosen from the following list for low correlation with other MOPs on the list and validity as measures of sensor use and effectiveness.

- Number of Tracks Initiated by Blue
- Percent of Red Forces Tracked
- Range, Time of First Blue Track File Generated

Summary

The ICAAS program has been developing innovative design concepts to assist tactical fighter pilots in defeating larger forces of enemy aircraft. Avionics, guidance, and control functions are being improved and extensively integrated to enhance pilot awareness of the combat situation, provide decision aiding in evaluating offensive/defensive options, support the pilot with precise execution of his engagement decisions, and coordinate the actions of individual flight members into an effective battle team. Efficient information management made possible by high throughput RISC based computers, combined with advanced cockpit display concepts, allow ICAAS to effectively support the pilot in achieving mission objectives.

The integration of the ICAAS system components was a very challenging task. The system development and integration was aided significantly by the use of the Air Combat Engagement System (ACES) and the use of Rapid Prototyping (RP) workstations. Extensive use of the MCAIR Advanced Integrated

Controls laboratory was made in preparing the ICAAS hardware/software for simulation evaluation and flight test verification.

A unique feature of the ICAAS manned simulations was the use of flightworthy computers as an integral part of the manned simulations right from the beginning. This was a necessity since these flightworthy computers were the only ones with sufficient throughput to execute the ICAAS code. This type of simulation configuration is more difficult to initially setup and check out. It requires iterative problem solving between the simulation facility and the integration laboratory. However, once checked out, it provides a significant advantage in that the software is now integrated into the computers required for flight test.

All ICAAS computer (IIC/ISC) software was programmed in Ada. There are over 850,000 32-bit words of executable code. We believe that the use of a higher order language such as Ada is the only practical way to develop so much software in a reasonable amount of time. It allows the system designers to develop the algorithm in the same language which is then subsequently tested in the laboratory, the simulator and in flight test. It would be prohibitively expensive to program ICAAS system computers in assembly language for simulation and flight test.

The piloted simulation evaluation is occurring in three major sessions. The first two consists of two blue aircraft versus up to eight threat aircraft, involve two simulation cockpits and are occurring in 1991-1992. The third session consists of four blue aircraft against up to 16 threat aircraft, involves four simulation cockpits and is scheduled to occur in 1992-1993. The measures of performance data generated in the simulations will be analyzed to assess ICAAS system tactical effectiveness along with ICAAS subsystem performance as it relates to situation awareness, flight management utilization, weapon utilization and sensor effectiveness.

References

1. Halski, D, Landy R, and Kocher J., "Integrated Control and Avionics for Air Superiority: A Knowledge-Based Decision-Aiding System, AGARD 51st Symposium of the Guidance and Control Panel, Madrid, September 1990.

2. Whitmoyer, R., "Techniques for Evaluating the Contributions of Avionics Subsystems to Fighter Aircraft Operational Effectiveness", IEE/AIAA Digital Avionics Systems Conference, Los Angeles, Oct. 1991.

From an Automated Flight-Test Management System to a Flight-Test Engineer's Workstation

E.L. Duke
R.W. Brumbaugh*
M.D. Hewett**
D.M. Tartt**
NASA Dryden Flight Research Facility
P.O. Box 273
Edwards, California 93523-0273

1 SUMMARY

This paper describes the capabilities and evolution of a flight-test engineer's workstation (called TEST_PLAN) from an automated flight-test management system. The concept and capabilities of the automated flight-test management system are explored and discussed to illustrate the value of advanced system prototyping and evolutionary software development.

2 NOMENCLATURE

ART	automated reasoning tool
ATMS	automated flight-test management system
dof	degree of freedom
FTE	flight-test engineer
FTTC	flight-test trajectory controller
FTTG	flight-test trajectory guidance
GUI	graphical user interface
HARV	High Alpha Research Vehicle
RDBMS	relational database management system
TACT	tactical aircraft technology

3 INTRODUCTION

This paper describes the development and capabilities of a flight-test engineer's workstation called TEST_PLAN and its evolution from the automated flight-test management system (ATMS). The ATMS was a tool for flight-test planning and scheduling; it contained expert systems for maneuver ordering, range management, and maneuver requirements evaluation. These expert systems were combined with three and six-degree-of-freedom simulations, state-of-the-art trajectory optimization, and a powerful graphic user interface to provide a desk top workstation.

TEST_PLAN is a computer program designed to run on standard graphics workstations as an aid to flight-test engineers (FTEs) in planning and executing flight-test programs. TEST_PLAN allows the FTE to organize and file extensive amounts of planning data while satisfying planning requirements on a flight-by-flight basis using aircraft and flight-specific information about instrumentation, telemetry, range, center-of-gravity, airborne and ground support, aerodynamic configuration, system configuration, and payload.

TEST_PLAN is the result of several generations of evolution. Originally combined with a maneuver autopilot, the first version of the ATMS was designed for flight-test maneuver planning and scheduling as well as maneuver execution and real-time flight-test monitoring; this first version of the ATMS was demonstrated in October 1987 using the NASA simulation facility at the Dryden Flight Research Facility. A second workstation version of ATMS evolved from lessons learned from the preliminary version—this second version eliminated the maneuver autopilot concept but retained a real-time flight monitoring capability; version two of the ATMS was demonstrated in mid-1990 at NASA using the F-18 High Alpha Research Vehicle (HARV) flight-test plans. A third commercial version of ATMS (called TEST_PLAN) resulted from the earlier experience and is designed as a FTE aid in planning and executing flight-test programs; TEST_PLAN is currently being used or considered for use by United States and international flight-test organizations.

4 THE AUTOMATED FLIGHT-TEST MANAGEMENT SYSTEM

The ATMS was originally developed at the NASA Dryden Flight Research Facility as a part of the NASA Aircraft Automation Program–a program focused on applying interdisciplinary state-of-the-art technology in artificial intelligence, control theory, and systems methodology to problems of operating and flight testing high-performance aircraft. In this section we present the background and a description of the ATMS [1,2,3].

*PRC Inc., Edwards, California
**G&C Systems Inc., San Juan Capistrano, California

4.1 Background of the Automated Flight-Test Management System

The ATMS was an outgrowth of the flight-test trajectory guidance (FTTG) work performed over the past decade on such programs as the F-111 Tactical Aircraft Technology (TACT) Program, the F-15 Propulsion/Airframe Integration Program, and the F-15 10°-Cone Program [4]. The FTTG provided display information to the pilot to allow complex, demanding flight research maneuvers to be flown more accurately. The FTTG was extended to a closed-loop system for the Highly Maneuverable Aircraft Technology (HiMAT, Program flight-test maneuver autopilot (FTMAP) [5]. In conjunction with this flight research at Dryden, Integrated Systems, Inc., under contract to NASA has developed a design methodology for these types of controllers [6,7,8] which has resulted in the basis of a flight-test trajectory controller (FTTC) which was flight tested in early 1990 on the F-15 Highly Integrated Digital Electronic Control (HIDEC) aircraft [9]. This FTTC was a major component of the ATMS as originally conceived and implemented.

The ATMS project was structured around a flight-test scenario and was an extension of work performed by SPARTA, Inc., (SPARTA, Inc., Laguna Hills, CA) under contract to NASA defining the need for a National Remote Computational Flight Research Facility (NRCFRF). The work on the NRCFRF contract defined the need for an expanded remotely augmented vehicle (RAV) capability and a flight program to demonstrate that capability. In the ATMS, a range, energy, and flight-test monitor expert system was used in conjunction with the FTTC to order maneuvers by priorities and energy management considerations while restricting the vehicle to the confines of a specified Edwards AFB test range. This expert system could be used online to control the research aircraft in flight and monitor the progress of a flight test; or offline as a planning tool for ordering the test maneuvers for a flight. The expert system used predictions of maneuvers based on simulation models for planning and actual flight-test data measurements for real-time vehicle control, data monitoring and flight test management.

4.2 Components of the Automated Flight-Test Management System

The main components of the ATMS were a trajectory controller based on the FTTC system [7,8], a flight-test planning expert system, a man–machine interface, and a flight-test monitoring expert system. The partitioning of functions in the ATMS was designed with two goals in mind; minimizing the bandwidth of the communication between components, and appropriate distribution of functions between numeric and symbolic processing.

The components described in this section perform the flight planning and monitoring functions. The fully developed ATMS (Fig. 1) was expected to perform program planning, block planning, and in-flight replanning which are not described herein as they were never implemented in the first ATMS.

4.2.1 Trajectory Controller

The trajectory controller was a collection of outer-loop guidance control laws which provide precise control for a

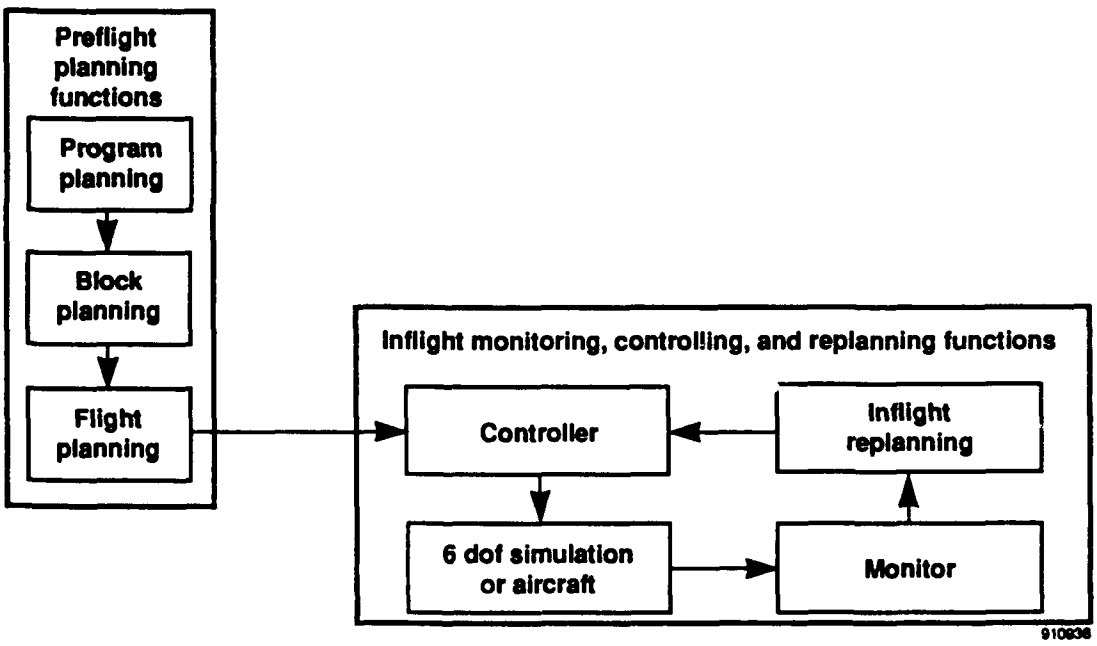


Fig. 1 ATMS functions.

vehicle performing high-quality flight research maneuvers such as level accelerations, wind-up turns, and pushover-pullup maneuvers. The trajectory controller was algorithmic, implemented in FORTRAN 77, and executed on a numeric processor.

The interface between the trajectory controller and the remaining components of the ATMS was designed to minimize the bandwidth of the communications across that interface. The trajectory controller accepted input commands consisting of an ordered list of maneuvers by type. Each maneuver consisted of a trim point, maneuver conditions, and end conditions. These commands contained from three to seven parameters each.

Once maneuver commands were received by the trajectory controller, the controller operated independently of the ATMS until another command list was received. The trajectory controller generated trajectories and trajectory following controls based on the maneuver commands and the aircraft instrumentation.

4.2.2 Flight-Test Planning Expert System

Flight-test planning must be done at several levels. At the highest level, the flights required for an entire program are established by the project requirements. At the next level, blocks of flights are determined by a more detailed analysis of the project requirements and are partitioned according to similarity of prerequisites, flight envelope requirements, and test needs to establish an orderly progression of blocks of flights satisfying the high-level project requirements. Within each block a number of individual flights are identified based on the detailed analysis of maneuvers required to satisfy the block requirements. Individual flights are then identified with a number of these maneuvers and the FTE must order maneuvers within a flight based on considerations of range, fuel, and energy management, as well as maneuver priorities.

The ATMS implemented only the test planner expert system. The test planner accepted a list of maneuvers and ordered them using rules that considered maneuver priorities, energy management, test range boundaries, and envelope limitations. Maneuvers which could not be included in the flight plan were eliminated from the plan being developed.

The flight-test planning expert system accepted test plan inputs from the FTE using a menu driven and icon based man–machine interface or previously stored test plan entries. When the list of test maneuvers was entered into the ATMS, the FTE selected the flight-test planning expert system which then used its knowledge base to order maneuvers, prioritize maneuvers, and construct a trajectory. As each maneuver was added to the planned trajectory, it was tested to insure that no system constraints had been violated. When constraint violations occurred, the flight-test planning expert system displayed information to the FTE describing the constraint violations and provided an explanation of the constraint, if requested. Maneuver priority was extremely important when fuel constraints were tested; lower priority maneuvers were removed from the test plan to satisfy fuel constraints.

The flight-test planning expert system was developed using the Automated Reasoning Tool (ART) expert system development environment hosted on a symbolic computer with a numeric processor board. It contained over 200 rules.

4.2.3 Man–Machine Interface

The man–machine interface component of the ATMS provided a means of information entry and display. This interface was used during flight planning and flight plan execution. The main display had three major components: the map, timeline, and command menu. In the map section of the main display were two types of displays: the trajectory planning display (Fig. 2), and the trajectory map display (Fig. 3). These map displays presented a two-dimensional view of the test range with the aircraft trajectory superimposed. The stored map was larger than the portion presented on the display. Pan and scroll were accomplished by using the mouse to choose an appropriate button depicted across the top of the display. A "navigate" button was also included to quickly determine course and distance between present aircraft position and any point within the stored map. The timeline component of the main display presented information on the aircraft trajectory in terms of altitude as a function of time or events. Figure 4 shows a timeline display of altitude as a function of time. Timeline scroll buttons allowed the FTE to examine different time or event segments by scrolling the timeline. The command menu portion of the main display allowed the user to select (using "mouse" or keyboard inputs) ATMS operational modes, maneuvers, or explanations of ATMS actions.

The man–machine interface was rule based with over 200 rules and presented on the computer monitor and keyboard. The interface was developed in ART.

4.2.4 Flight-Test Monitor Expert System

The flight-test monitor expert system provided an interface between the FTE and either the planned trajectory or the actual trajectory (whether generated by simulation or flight). This system also provided the trajectory controller with inputs from the list of maneuvers in the planned trajectory.

The flight-test monitor expert system issued maneuver requests to the trajectory controller, then monitored the aircraft parameters of interest to insure that no system constraints were violated. This system also monitored maneuver quality. When a system constraint was violated or the quality of a maneuver was unacceptable, the flight-test monitor expert system notified the FTE of the problem and made recommendations based on the information within its knowledge base. Each maneuver was selected from the list of planned maneuvers in order; the flight-test monitor

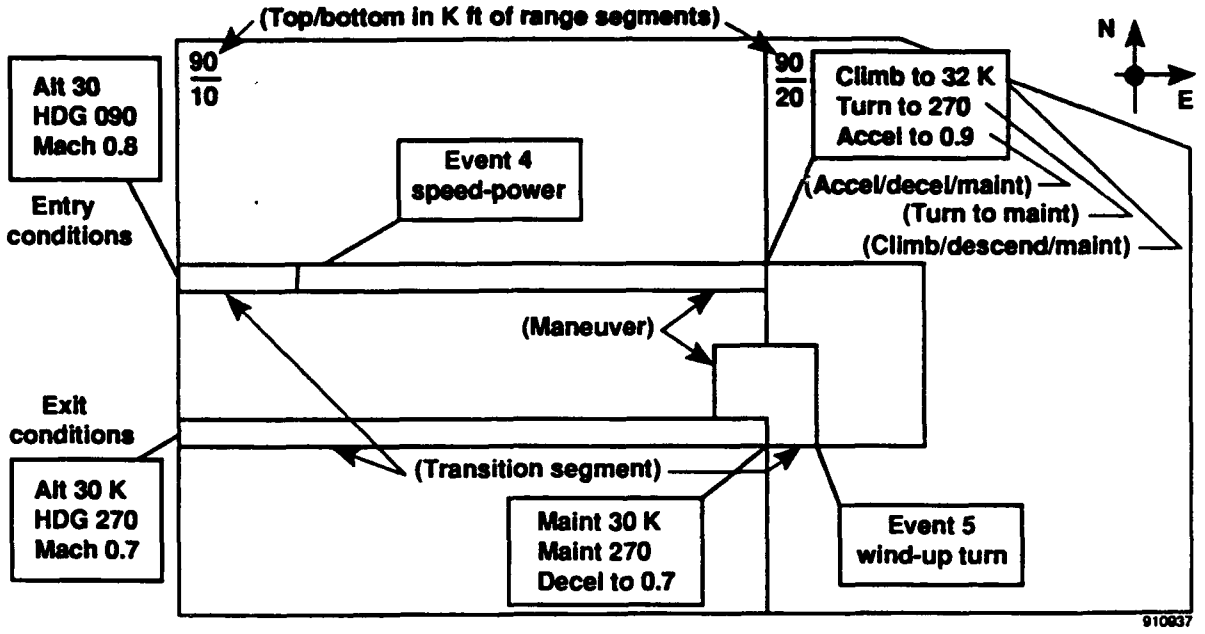


Fig. 2 Trajectory planning display.

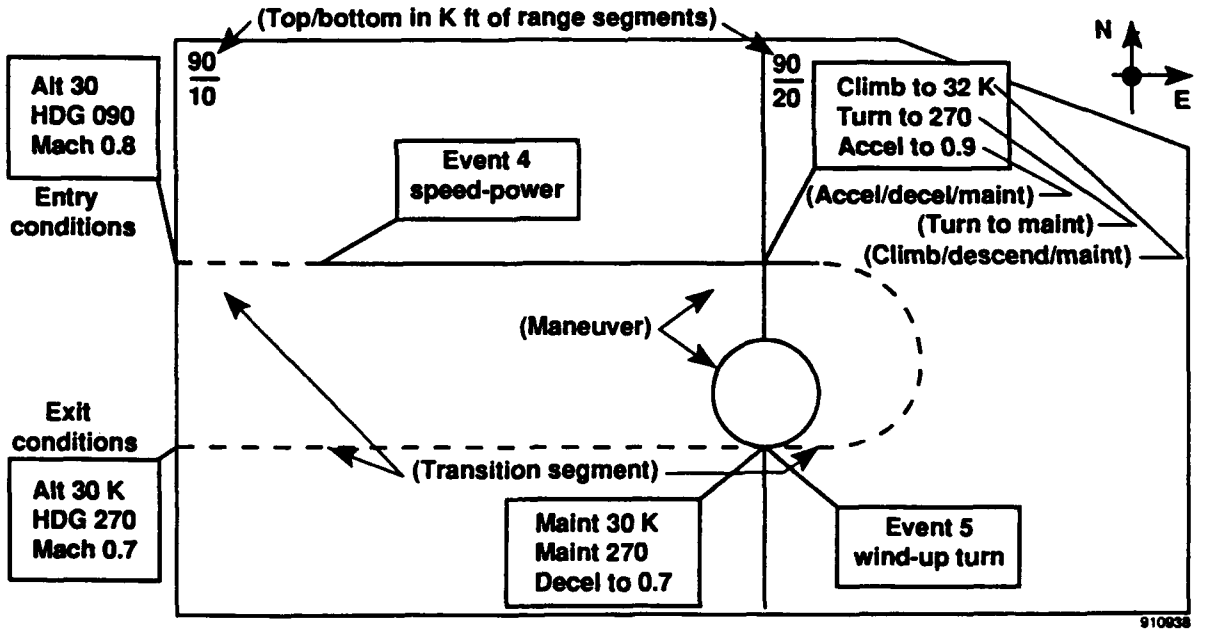


Fig. 3 Trajectory map display.

expert system initiated these maneuvers and then waited for the trajectory controller to finish a maneuver before proceeding to the next maneuver on the list.

4.3 Automated Flight-Test Management System Configurations

The ATMS had three configurations: the FTE workstation, the simulation validation system, and the flight system. The FTE workstation and the simulation validation system were used to develop and evaluate flight-test plans. The simulation validation system was also used to aid in the validation of the flight system including aircraft modifications. The flight system was used to actually conduct flight test by executing the flight-test plan, monitoring the performance of the aircraft, and controlling the aircraft in flight.

4.3.1 Flight-Test Engineer's Workstation

The configurations of the FTE workstation is shown in Figure 5. This system was used by the FTE to develop

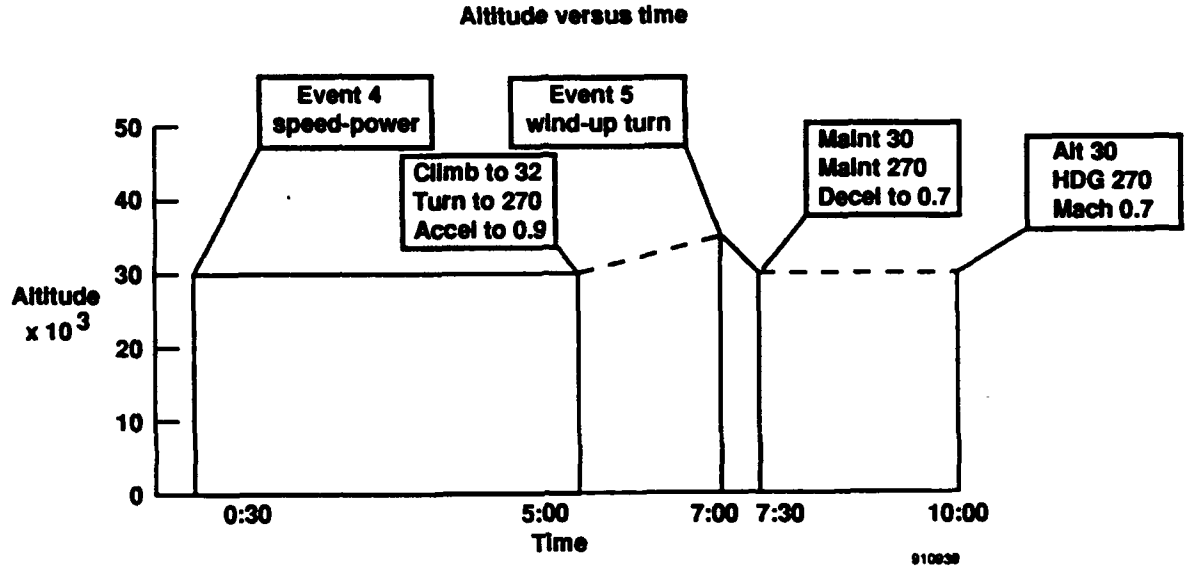


Fig. 4 Timeline display.

preliminary flight-test plans without having to use the aircraft simulator. This provided the FTE with a standalone system that was separated from the aircraft simulator, which was always in great demand, and thus allowed more flexibility in test plan development.

The FTE workstation included two computers; a symbolic computer with a numeric processor board and a graphics workstation. The LISP processor on the computer contained the flight-test planning expert system, the man–machine interface system, and the rule-based portion of the flight-test monitoring expert system. The numeric processor on the symbolic computer contained a three degree-of-freedom (3 dof) digital performance simulation (DPS) and the software to execute the algorithmic, trajectory management portion of the flight-test management expert system. The LISP processor and numeric processor board communicated using an internal bus. The graphics workstation contained a six degree-of-freedom (6 dof) simulation of the aircraft and the FTTC. The two computers communicated using Ethernet with a standard protocol.

4.3.2 Simulation Validation System

The configuration of the simulation validation system is shown in Figure 6. This system was used by the FTE to evaluate flight plans developed on the FTE workstation to provide detailed pilot-in-the-loop mission briefing and familiarization, and as a validation facility for testing the ATMS as well as the ground and aircraft systems to be used in the actual flight testing.

The simulation validation configuration of the ATMS included three computers; the symbolic computer and two real-time mini computers. The computer in the simulation validation system was configured identically to the FTE workstation configuration of this processor. One mini computer (designated the "control law computer") contained the trajectory controller software and communicated with the symbolic computer using a standard protocol. The communication between this mini computer and the symbolic computer was identical to the communication between the computer and the graphics workstation in the FTE workstation configuration. The other mini computer contained a detailed 6 dof simulation of the aircraft and also contained detailed models of the downlink and uplink telemetry system. The two mini computers communicated in engineering units through FORTRAN named common blocks using a two-port shared memory.

4.3.3 Flight System

The configuration of the ATMS flight system is shown in Figure 7. The flight system was to be used to conduct flight test by executing the flight test plan, monitoring the performance of the aircraft, and controlling the aircraft in flight.

The flight system configuration of the ATMS included three computers; a symbolic computer and two mini computers. The computer in the flight system was configured identically to the FTE workstation and simulation validation system configurations of this computer. One mini control law computer contained the trajectory controller software and communicated with the computer using a standard protocol. The communication between the control law computer and the smaller computer was identical to the communication between the smaller computer and the control law computer in the simulation validation system configuration. A second mini computer (designated the "engineering units computer") was included in the flight system and provided processing required for the uplink and downlink telemetry systems. The communication between

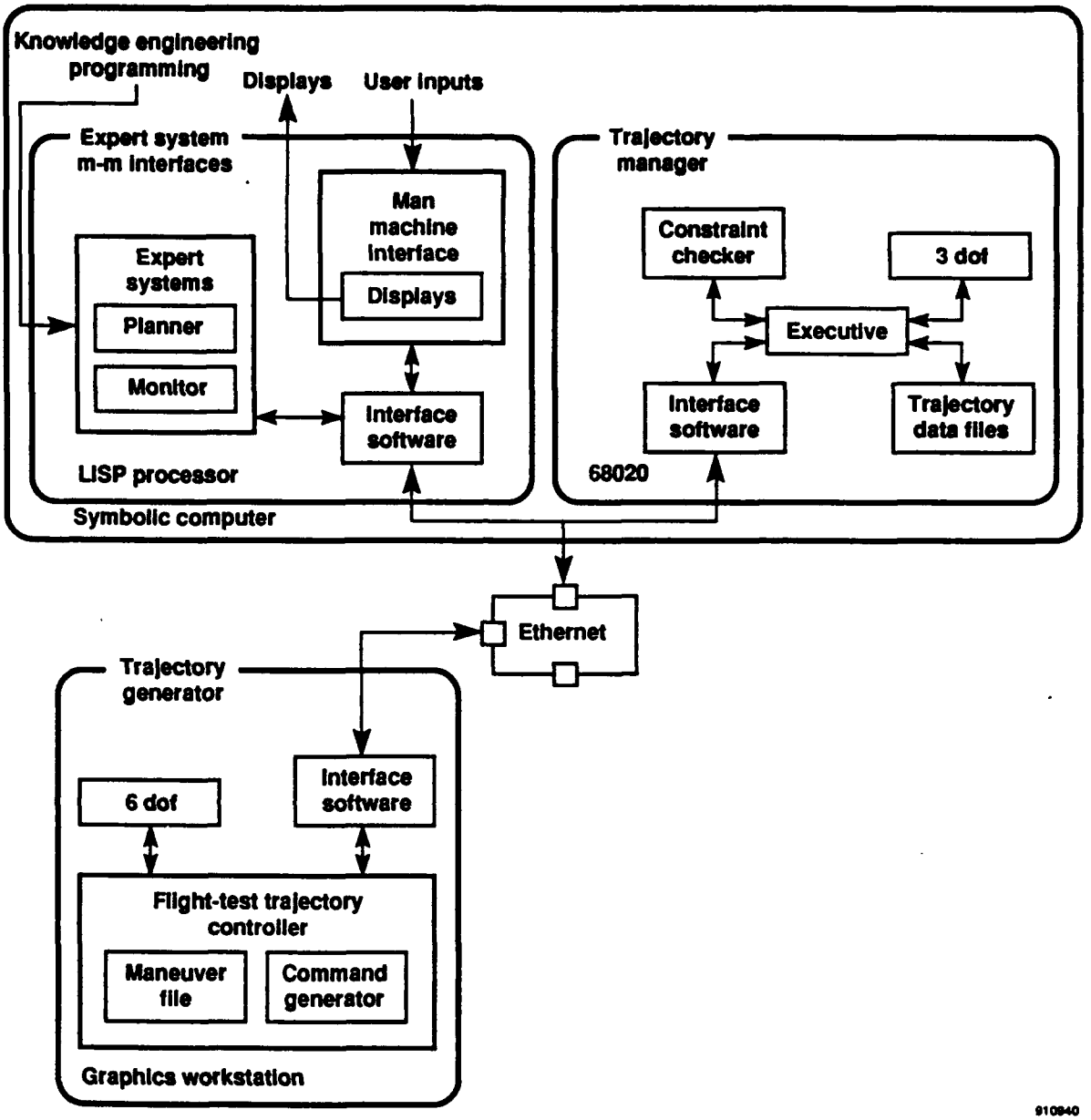


Fig. 5 FTE workstation configuration.

the two mini computers was identical to the communication between the two mini computers in the simulation validation configuration. In the flight system, the simulation model of the aircraft and telemetery systems were to be replaced with actual systems.

5 THE EVOLUTION OF TEST_PLAN FROM THE AUTOMATED FLIGHT-TEST MANAGEMENT SYSTEM

The first version of the ATMS was used to develop the rapid prototyping facility for flight research in advanced systems concepts [2,10]. This rapid prototyping facility was intended to allow easy transition from concept to simulation then to flight. Not only was ATMS the first system to be used in this facility, it was the first system to benefit from the capabilities provided by this facility.

As originally conceived, ATMS was to have combined several concepts into a single system that would allow planning, simulation, execution, and monitoring of research test flights. But this was unachievable for several reasons. The most apparent problem was the inadequacies of the symbolic processors and the expert system development language when applied to real-time tasks. Without this

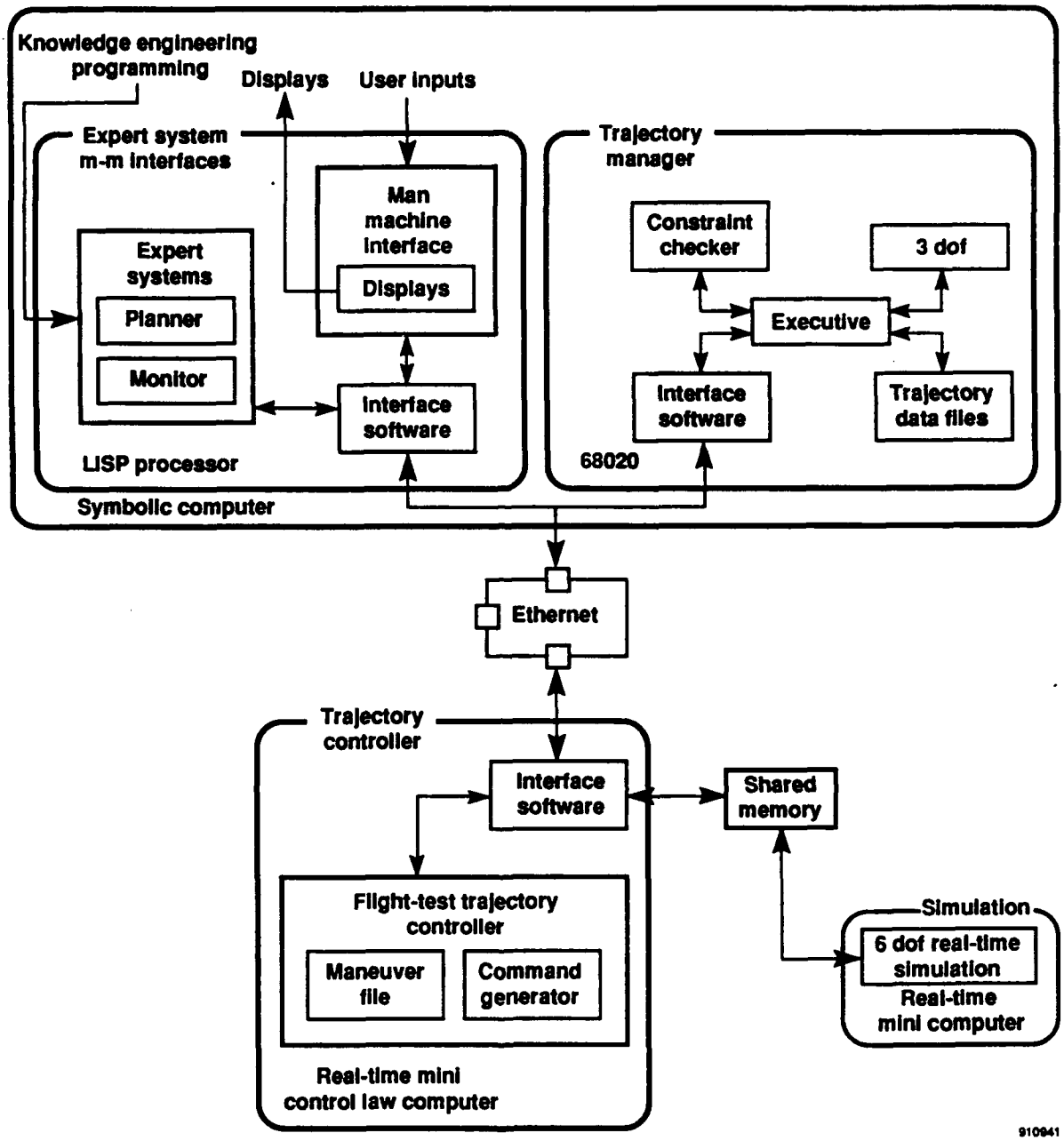


Fig. 6 Simulation validation system configuration.

facility these problems might not have been detected until much later in the development program.

The problem of computers and expert system development languages was addressed by re-implementing the expert systems using CLIPS ('C' Language Production System), converting from LISP to 'C,' using X-windows for the graphical user interface, and re-hosting the system on standard numeric workstations.

Finally, experience in the rapid prototyping showed the difficulties inherent in a system as ambitious as ATMS. Instead of a single system to manage all aspects of planning, simulation, execution, and monitoring, we decided to develop several separate but compatible systems. Thus, the rapid prototyping facility allowed realistic decisions to be made about the viability of the ATMS concept.

TEST_PLAN is the result of the decision to expand the portion of ATMS that provided the FTE with a planning tool. This system, while the development was under government auspices, was called the "flight test engineer's workstation" [3] and TEST_PLAN when extended and commercialized by G & C Systems Inc. (G & C Systems Inc., San Juan Capistrano, CA).

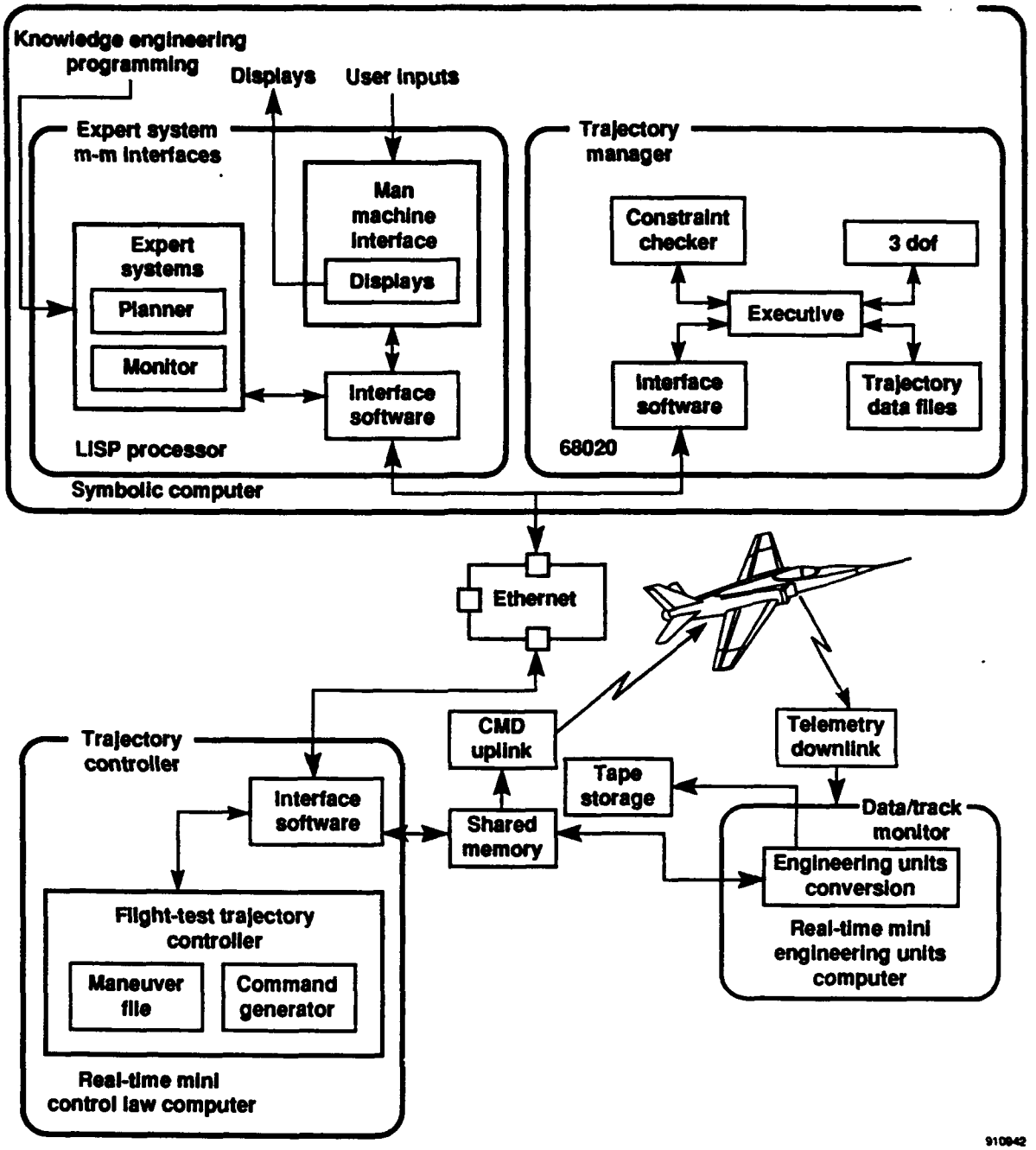


Fig. 7 Flight system configuration.

6 TEST_PLAN

TEST_PLAN is a computer program designed to run on standard graphics workstations (under either the UNIX® or VMS operating systems) as an aid to FTEs in planning and executing flight-test programs. TEST_PLAN allows the FTE to organize and file extensive amounts of planning data while satisfying planning requirements on a flight-by-flight basis using aircraft and flight specific information about instrumentation, telemetry, range, center-of-gravity, airborne and ground support, aerodynamic configuration, system configuration, and payload.

6.1 TEST_PLAN Components

The primary components of TEST_PLAN (shown in Fig. 8) include:

1. A planning facility with over 1000 flight-test planning procedures,

® UNIX is a registered trademark of AT & T Bell Laboratories, Whippany, New Jersey.

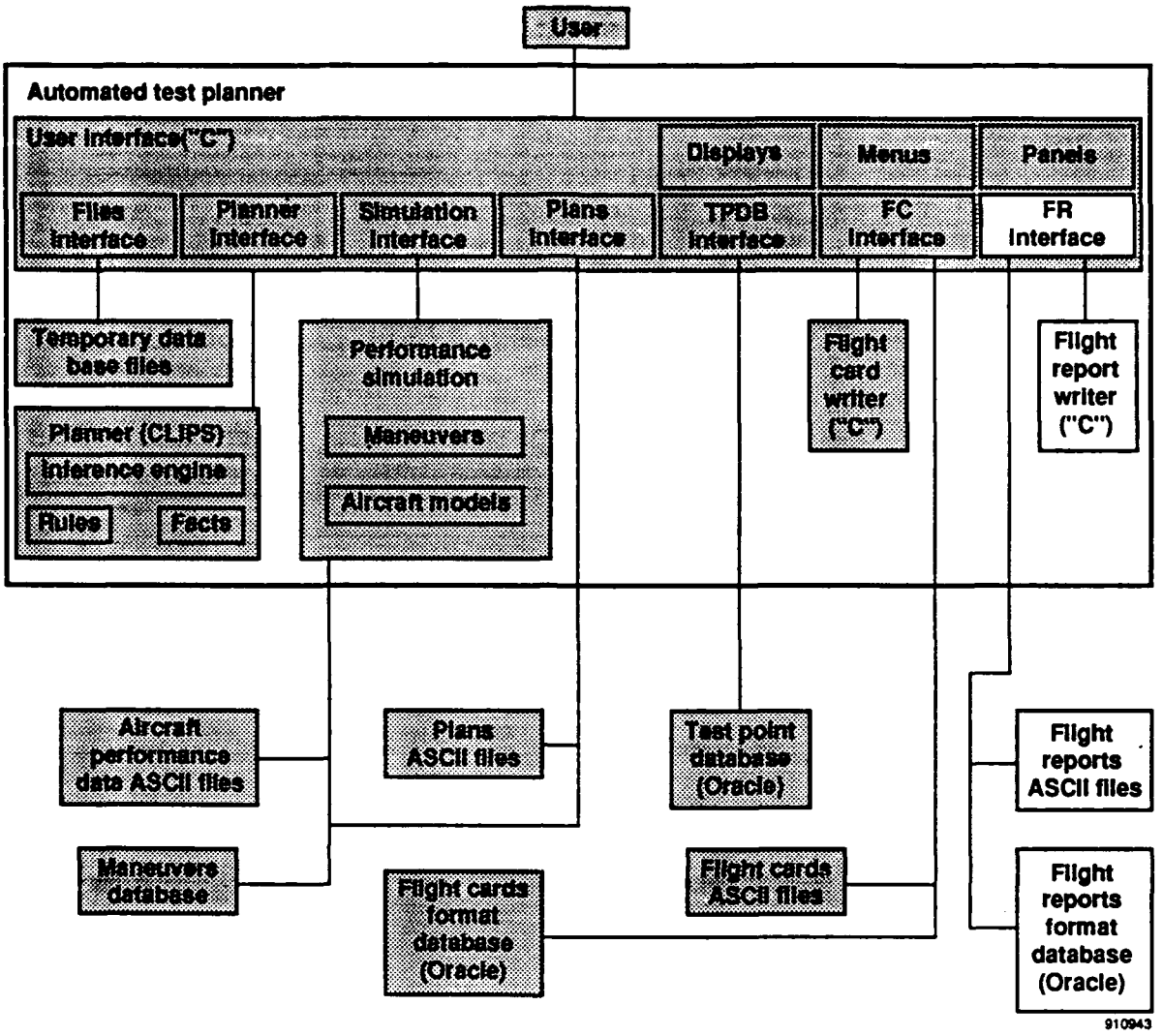


Fig. 8 TEST_PLAN system architecture.

2. an extensive graphical user interface (GUI),
3. an interface with a relational database management system (RDBMS),
4. an aircraft performance simulation facility,
5. a flight card generation facility, and
6. expert system based planning aids.

In the following sections, we will discuss these components of TEST_PLAN.

6.1.1 Planning Facility

The heart of TEST_PLAN [11] is a planning facility consisting of a planning matrix and over 1000 planning procedures. The planning matrix consists of flight-test maneuvers and contingency maneuvers organized by flights for each individual aircraft in a flight-test program. The planning matrix is displayed in an easy to use format (Fig. 9).

The automated planning procedures allow the FTE to plan flight-test programs by defining maneuvers and filling out the planning matrix. The basic philosophy implemented in the TEST_PLAN planning facility (Fig. 10) features the creation of a centralized database of test points in an RDBMS which must be satisfied in the flight-test program; the creation of multiple flight-test plans consisting of test points assigned to flight-test maneuvers, flight-test maneuvers assigned to flights, and flights assigned to blocks of flights for specific flight-test aircraft; and continuous, automated constraint checking between test points, maneuvers, and flights for test point requirements and flight assignments on a flight-by-flight basis.

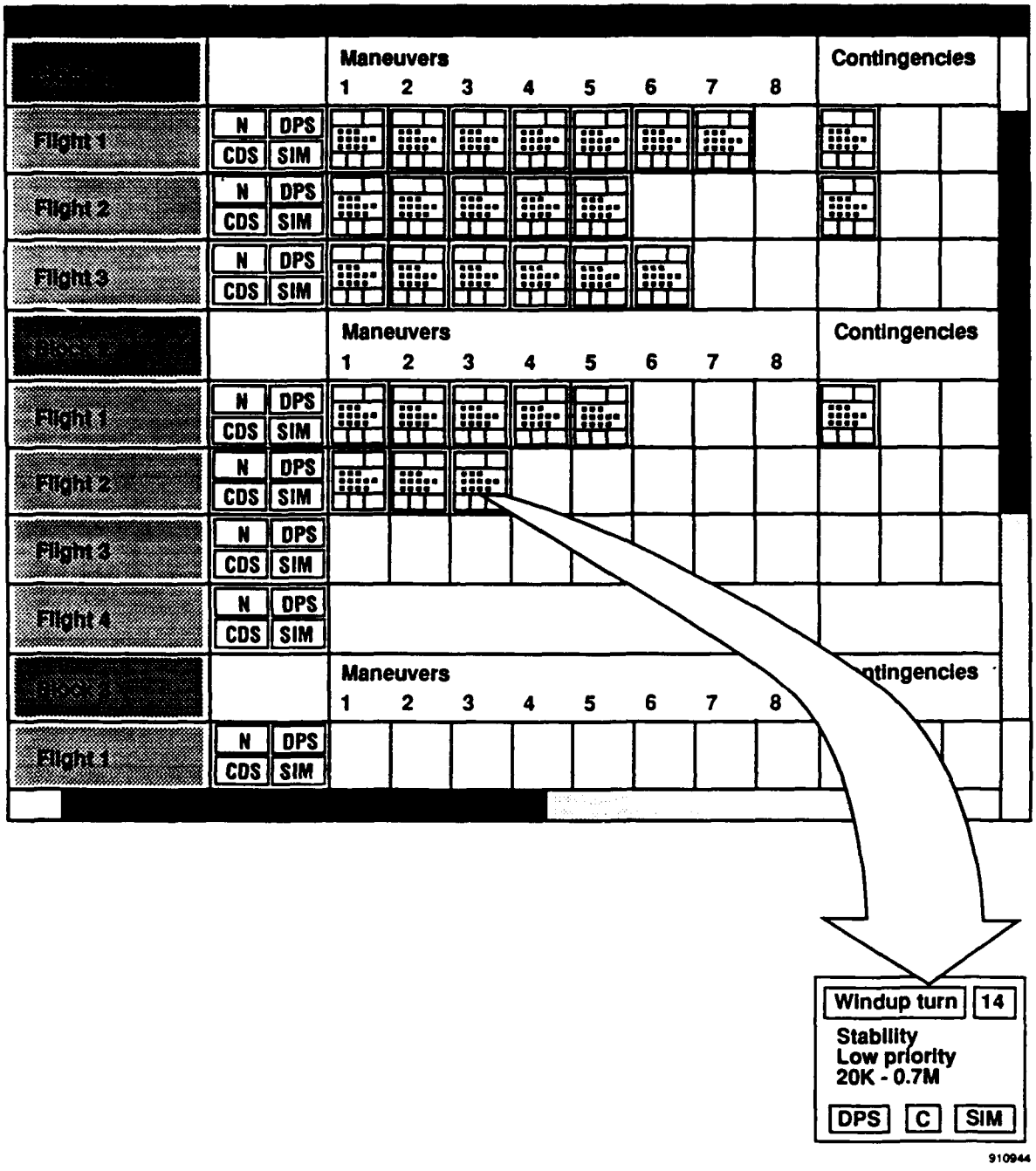


Fig. 9 Planning matrix display.

6.1.2 Graphical User Interface

TEST_PLAN uses graphics extensively. One of its most visible features is a highly developed GUI using X-windows. This interface consists of windows, panels, canvasses, action buttons, and menus. Maximum use of is made of mouse initiated operations. The interface permits the FTE to execute procedures in any order desired—the FTE is not limited to the serial, predefined order of events typically found in a menu-driven application.

Using the TEST_PLAN GUI, the flight-test engineer can perform many tasks which would normally require extensive paper and pencil work. These automated tasks include laying out planning matrices, planning blocks of flights, defining test points, defining flight-test maneuvers, assigning test points to flight-test maneuvers, sequencing flight-test maneuvers to minimize fuel and time required, writing flight cards, and satisfying test point constraints. These test points constraints may be based on instrumentation, aircraft configuration, range (operating area) requirements, telemetry requirements, system configuration,

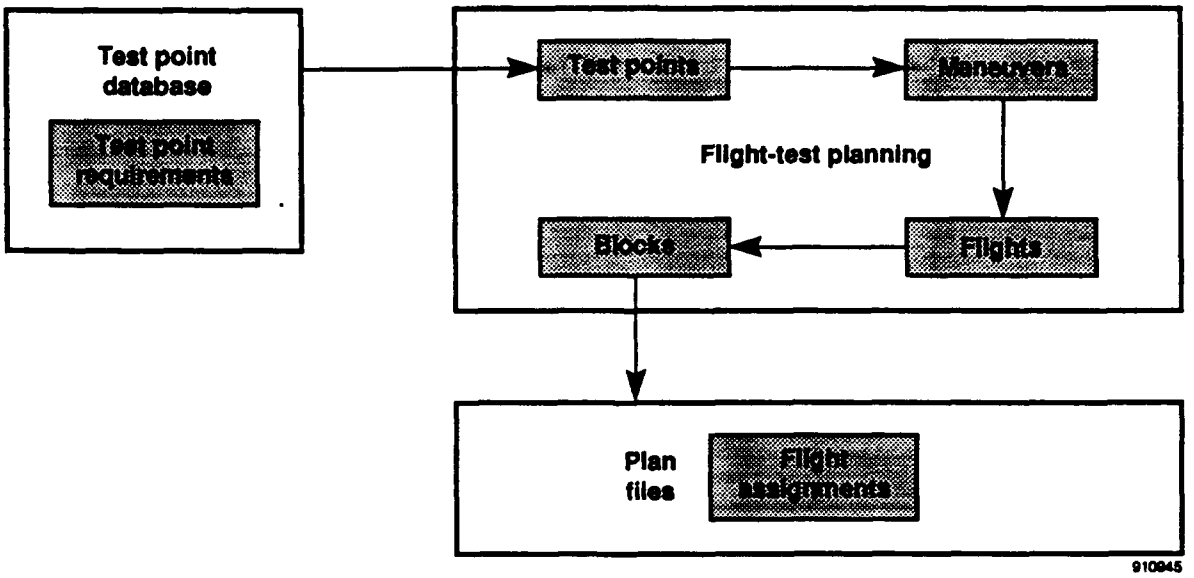


Fig. 10 The TEST_PLAN planning philosophy.

air and ground support requirements, payload, weight and balance requirements, flight limits, trim point conditions, or test point prerequisites.

6.1.3 Relational Database Management System

TEST_PLAN is functionally integrated with the Oracle Relational Database Management System (RDBMS). The RDBMS contains a database of test points for a flight-test program. TEST_PLAN incorporates many procedures which greatly simplify database queries, record additions, modifications and deletions. No special knowledge of the database query language is required because TEST_PLAN provides the user a set of menus, action buttons, and data entry fields as part of the GUI.

6.1.4 Aircraft Performance Simulation

TEST_PLAN contains a 3 dof generic aircraft performance simulation. This simulation requires the user to define an aerodynamic model (of lift and drag coefficients as functions of Mach number and angle of attack) and a propulsion system model (of thrust and fuel flow as functions of altitude and power lever angle).

Using the simulation and this simple definition of the vehicle, TEST_PLAN can compute the trajectory and fuel used in any of 52 preprogrammed maneuvers such as climbs, descents, level accelerations and decelerations, cruise, turns, and dynamic maneuvers. The flight-test engineer also has the capability of building new maneuvers by stringing together combinations of individual maneuvers.

6.1.5 Flight Card Generation Facility

TEST_PLAN provides a flight card generation facility which uses default entries from the flight card database to generate a set of flight cards for a specific flight. The nominal flight card is shown in Figure 11. However, the format can be customized to any desired during the customization portion of an installation of TEST_PLAN.

6.1.6 Expert System Based Planning Aids

TEST_PLAN contains two expert system planning aids; a flight planner and a block planner. The block planner assigns maneuvers (which contain test points) to flights, attempting to minimize the number of flights required to execute the maneuvers within the block while satisfying constraints on instrumentation, configuration, flight limits, flight conditions, prerequisite test points, and range requirements. The flight planner reorders maneuvers within individual flights attempting to satisfy constraints while minimizing fuel and range time used; the flight planner uses fuel and time data obtained from trajectories generated in the performance simulation. An explanation facility is provided.

7 CONCLUDING REMARKS

This report describes the automated flight-test management system and an automated flight test planning system called TEST_PLAN. The evolution of TEST_PLAN from automated flight-test management system is detailed to illustrate the use of rapid prototyping to define system requirements.

POS	PM	EXP	H/Mach		Card number

910946

Fig. 11 Nominal flight card format.

8 REFERENCES

[1] Hewett, Marle D., David M. Tartt, Eugene L. Duke, Robert F. Antoniewicz, and Randal W. Brumbaugh, "The Development of an Automated Flight Test Management System for Flight Test Planning and Monitoring," Proceedings of the First International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Tullahoma, TN, June 1–3, 1988, Vol. 1, University of Tennessee, 1988, pp. 324–333.

[2] Duke, Eugene L., Marle D. Hewett, Randal W. Brumbaugh, David D. Tartt, Robert F. Antoniewicz, and Arvind K. Agarwal, "The Use of an Automated Flight Test Management System in the Development of a Rapid-Prototyping Flight Research Facility," 4th Conference on Artificial Intelligence Applications, Long Beach, CA, May 4–6, 1988. (See also NASA TM-100435, 1988.)

[3] Tartt, David D., Marle D. Hewett, Eugene L. Duke, James A. Cooper, and Randal W. Brumbaugh, "The Development of a Flight Test Engineer's Workstation for the Automated Flight Test Management System," Proceedings of the Society of Flight Test Engineers, 20th Annual Symposium, Reno, NV, Sept. 18–21, 1989, pp. 5.2-1 to 5.2-12.

[4] Duke, Eugene L., Michael R. Swann, Einar K. Enevoldson, and Thomas D. Wolf, "Experience with Flight Test Trajectory Guidance," *J. Guidance, Control and Dynamics*, Vol. 6, No. 5, 1983, pp. 393–398.

[5] Duke, Eugene L., Frank P. Jones, and Ralph B. Roncoli, *Development and Flight Test of an Experimental Maneuver Autopilot for a Highly Maneuverable Aircraft*, NASA TP-2618, 1986.

[6] Walker, Robert and Naren Gupta, "Flight Test Trajectory Control Analysis," NASA CR-170935, 1983.

[7] Menon, P.K.A. and R.A. Walker, "Aircraft Flight Test Trajectory Control," ISI Report No. 56, Integrated Systems, Inc., Santa Clara, CA, Oct. 1985. (See also NASA CR-179428, 1988.)

[8] Menon, P.K.A., M.E. Badgett, and R.A. Walker, "Nonlinear Maneuver Autopilot for the F-15 Aircraft," ISI Report No. 71, Integrated Systems, Inc., Santa Clara, CA, Dec. 1985.

[9] Antoniewicz, Robert F., Eugene L. Duke, and P.K.A. Menon, "Flight Test of a Trajectory Controller Using Linearizing Transforms With Measurement Feedback," *AIAA Guidance, Navigation and Control Conference, Part 1*, Portland, OR, AIAA 90-3373, Aug. 1990, pp. 518–532.

[10] Duke, Eugene L., Randal W. Brumbaugh, and James D. Disbrow, "A Rapid Prototyping Facility for Flight Research in Advanced Systems Concepts," *Computer*, May 1989, pp. 61–66.

[11] *Automated Flight Test Planning System: TEST_PLAN User's Guide*, G & C Systems, Inc., 30250 Rancho Viejo Road, San Juan Capistrano, CA 92675, 1991.

REPORT DOCUMENTATION PAGE			
1. Recipient's Reference	**2. Originator's Reference**	**3. Further Reference**	**4. Security Classification of Document**
	AGARD-CP-504	ISBN 92-835-0662-6	UNCLASSIFIED

5. Originator	Advisory Group for Aerospace Research and Development North Atlantic Treaty Organization 7 rue Ancelle, 92200 Neuilly sur Seine, France
6. Title	AIR VEHICLE MISSION CONTROL AND MANAGEMENT
7. Presented at	the Guidance and Control Panel 53rd Symposium held at the Marine Kazerne, Amsterdam, The Netherlands from 22nd October to 25th October 1991.

8. Author(s)/Editor(s)	**9. Date**
Various	March 1992
10. Author's/Editor's Address	**11. Pages**
Various	270

12. Distribution Statement This document is distributed in accordance with AGARD policies and regulations, which are outlined on the back covers of all AGARD publications.

13. Keywords/Descriptors

Avionics	Fuzzy logic
Mission planning	Expert systems
Aircraft	Trajectories
Missiles	Guidance and control
Flight control	Integrated systems

14. Abstract

This volume contains the 21 unclassified papers (Paper no. 2 was not available at time of printing), presented at the Guidance and Control Panel Symposium held at the Marine Kazerne, Amsterdam, The Netherlands from 22nd to 25th October 1991.

The papers were presented covering the following headings:

- Operational Mission Considerations;
- Situation Assessment;
- Route Planning;
- Planning Techniques;
- Implementation Aspects.

AGARD Conference Proceedings 504
Advisory Group for Aerospace Research and Development, NATO
AIR VEHICLE MISSION CONTROL AND MANAGEMENT
Published March 1992
270 pages

This volume contains the 21 unclassified papers (Paper no. 2 was not available at time of printing), presented at the Guidance and Control Panel Symposium held at the Marine Kazerne, Amsterdam, The Netherlands from 22nd to 25th October 1991.

P.T.O.

AGARD-CP-504

Avionics
Mission planning
Aircraft
Missiles
Flight control
Fuzzy logic
Expert systems
Trajectories
Guidance and control
Integrated systems

AGARD Conference Proceedings 504
Advisory Group for Aerospace Research and Development, NATO
AIR VEHICLE MISSION CONTROL AND MANAGEMENT
Published March 1992
270 pages

This volume contains the 21 unclassified papers (Paper no. 2 was not available at time of printing), presented at the Guidance and Control Panel Symposium held at the Marine Kazerne, Amsterdam, The Netherlands from 22nd to 25th October 1991.

P.T.O.

AGARD-CP-504

Avionics
Mission planning
Aircraft
Missiles
Flight control
Fuzzy logic
Expert systems
Trajectories
Guidance and control
Integrated systems

AGARD Conference Proceedings 504
Advisory Group for Aerospace Research and Development, NATO
AIR VEHICLE MISSION CONTROL AND MANAGEMENT
Published March 1992
270 pages

This volume contains the 21 unclassified papers (Paper no. 2 was not available at time of printing), presented at the Guidance and Control Panel Symposium held at the Marine Kazerne, Amsterdam, The Netherlands from 22nd to 25th October 1991.

P.T.O.

AGARD-CP-504

Avionics
Mission planning
Aircraft
Missiles
Flight control
Fuzzy logic
Expert systems
Trajectories
Guidance and control
Integrated systems

AGARD Conference Proceedings 504
Advisory Group for Aerospace Research and Development, NATO
AIR VEHICLE MISSION CONTROL AND MANAGEMENT
Published March 1992
270 pages

This volume contains the 21 unclassified papers (Paper no. 2 was not available at time of printing), presented at the Guidance and Control Panel Symposium held at the Marine Kazerne, Amsterdam, The Netherlands from 22nd to 25th October 1991.

P.T.O.

AGARD-CP-504

Avionics
Mission planning
Aircraft
Missiles
Flight control
Fuzzy logic
Expert systems
Trajectories
Guidance and control
Integrated systems

The papers were presented covering the following headings:

- Operational Mission Considerations;
- Situation Assessment;
- Route Planning;
- Planning Techniques;
- Implementation Aspects.

ISBN 92-835-0662-6

The papers were presented covering the following headings:

- Operational Mission Considerations;
- Situation Assessment;
- Route Planning;
- Planning Techniques;
- Implementation Aspects.

ISBN 92-835-0662-6

The papers were presented covering the following headings:

- Operational Mission Considerations;
- Situation Assessment;
- Route Planning;
- Planning Techniques;
- Implementation Aspects.

ISBN 92-835-0662-6

KEYNOTE ADDRESS

by

Major-General W. Breeschoten
Ministerie van Defensie (KLu)
Luchtmachstaf
P O Box 20703
2500 Es Den Haag
The Netherlands

I. INTRODUCTION

Ladies and Gentlemen,

In addition to my present function as Director Operations of the RNLAF I am a fighter pilot.

This week you will be discussing my job; the management and control of missions.
Therefore I was pleased, both with the subject of this symposium and with your kind invitation to address you at the beginning of this week.

In this address I will start with my views as a pilot on the subject of mission control support systems, (you already noted the word support) and work towards my views as Director. These views are the same, only the scope is different.

II. MISSION CONTROL AND THE PILOT

As a pilot, or group of pilots I get assigned a mission, say to attack a certain ground target with the objective to destroy it, or make it unusable for a certain period of time. My job then is to prepare and execute this mission.

My preparation consists largely of building up an image of what precisely is required and under what circumstances.
For this I collect information on the situation enroute, the situation at the target, the assistance that can be expected from supporting forces and when, the weather situation and so on.

Then the mission is planned, which concerns a multitude of items:
route selection, flight profile planning, weapons selection and attack profile planning, planning for mutual protection , planning for ECM employment, planning for threat system counters, not to mention planning for relative timing and required command and information exchanges with my companions with supporting units and with other missions.

Now I know what I can expect during my mission and how to manage possible situations. Also I know how these situations can be identified, using my on-board sensor data. For situations that car be anticipated I know what I am supposed to do and how to do it.

During execution I use the preparation results in conjunction with my system displays and my mkl eyeball to maintain an image of the situation, assess its meaning for my mission, decide on actions and of course carry them out.

This process, from preparation to execution, is called mission control. In the complex technical and operational environment of today and tomorrow I can do with some help when doing it.

First of all in the field of information during preparation. The more I know, the less surprises, and the larger the chance of a successful mission. Therefore information must be recent, reliable, and preferably accurate and complete. Most of all it must be easy to interpret.
For instance, I want to know what my target looks like now, not last week; where precisely it is or is expected to be, and what tricks can be expected in terms of decoys camouflage, defenses etc.

Then, when planning my mission, I certainly like support in clerical tasks.
Let a system calculate my fuel use for me, plot my route, identify the risk when engaging threats for the tactics selected, document my planning for me and prepare my system data loader for me.
Let it allow me to assess the consequences of an info update at the end of my planning process and make it possible to adapt the primary plan if necessary.

During my mission I would like my systems to show me the situation in an easily understood manner. Show me relevant information only. I could indicate premisssion what I consider relevant and what not. It can also show me my own capabilities in relation to threats.

You will notice that I stress the information supply side, and the improvement of the quality of information, to make it directly accessible for the pilot, without the need to interpret raw data.
I do not want the system to think for me, at least not in the sense that it prescribes my tactical actions.
It can to some extent think with me. For example to carry out some tasks that I do not wish to be bothered with; it may monitor my other systems, decide which ones do not perform correctly, re-configure, and inform me of any degradation in performance, not redundancy.
Also, if I want so, a system can perform specific tasks for me, for instance do my self-protection, activate and shut down jammers and throw decoys, provided its operation is transparent, which means that I know what it will be doing.

More in general, the functions of systems that support me need to be clear. I have to understand what their products mean.
For route planning I like a system that shows me the known threat, and indicates what possibly could be there that is unknown at present.
I certainly do not want a system that calculates an optimum route for me under the erroneous assumption that the world is known, with the result that I have to meander towards my target and then run into a few surprises all the same.

It is imperative that mission control support for manned systems is designed in such a manner thatit supports the pilot, or more in general the operator. If he can not maintain a clear image of the situation, he cannot perform his job. Also, if the support systems do perform part of his primary job, he does lack the freedom of control he may need. And if you let the system do it all, you do not need a pilot.

AGARD

NATO OTAN

7 RUE ANCELLE · 92200 NEUILLY-SUR-SEINE

FRANCE

Téléphone (1)47.38.57.00 · Télex 610 176
Télécopie (1)47.38.57.99

DIFFUSION DES PUBLICATIONS
AGARD NON CLASSIFIEES

L'AGARD ne détient pas de stocks de ses publications, dans un but de distribution générale à l'adresse ci-dessus. La diffusion initiale des publications de l'AGARD est effectuée auprès des pays membres de cette organisation par l'intermédiaire des Centres Nationaux de Distribution suivants. A l'exception des Etats-Unis, ces centres disposent parfois d'exemplaires additionnels; dans les cas contraire, on peut se procurer ces exemplaires sous forme de microfiches ou de microcopies auprès des Agences de Vente dont la liste suite.

CENTRES DE DIFFUSION NATIONAUX

ALLEMAGNE
Fachinformationszentrum,
Karlsruhe
D-7514 Eggenstein-Leopoldshafen 2

BELGIQUE
Coordonnateur AGARD-VSL
Etat-Major de la Force Aérienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Bruxelles

CANADA
Directeur du Service des Renseignements Scientifiques
Ministère de la Défense Nationale
Ottawa, Ontario K1A 0K2

DANEMARK
Danish Defence Research Board
Ved Idraetsparken 4
2100 Copenhagen Ø

ESPAGNE
INTA (AGARD Publications)
Pintor Rosales 34
28008 Madrid

ETATS-UNIS
National Aeronautics and Space Administration
Langley Research Center
M/S 180
Hampton, Virginia 23665

FRANCE
O.N.E.R.A. (Direction)
29, Avenue de la Division Leclerc
92320, Châtillon sous Bagneux

GRECE
Hellenic Air Force
Air War College
Scientific and Technical Library
Dekelia Air Force Base
Dekelia, Athens TGA 1010

ISLANDE
Director of Aviation
c/o Flugrad
Reykjavik

ITALIE
Aeronautica Militare
Ufficio del Delegato Nazionale all'AGARD
Aeroporto Pratica di Mare
00040 Pomezia (Roma)

LUXEMBOURG
Voir Belgique

NORVEGE
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25
N-2007 Kjeller

PAYS-BAS
Netherlands Delegation to AGARD
National Aerospace Laboratory NLR
Kluyverweg 1
2629 HS Delft

PORTUGAL
Portuguese National Coordinator to AGARD
Gabinete de Estudos e Programas
CLAFA
Base de Alfragide
Alfragide
2700 Amadora

ROYAUME UNI
Defence Research Information Centre
Kentigern House
65 Brown Street
Glasgow G2 8EX

TURQUIE
Milli Savunma Başkanlığı (MSB)
ARGE Daire Başkanlığı (ARGE)
Ankara

LE CENTRE NATIONAL DE DISTRIBUTION DES ETATS-UNIS (NASA) NE DETIENT PAS DE STOCKS DES PUBLICATIONS AGARD ET LES DEMANDES D'EXEMPLAIRES DOIVENT ETRE ADRESSEES DIRECTEMENT AU SERVICE NATIONAL TECHNIQUE DE L'INFORMATION (NTIS) DONT L'ADRESSE SUIT.

AGENCES DE VENTE

National Technical Information Service
(NTIS)
5285 Port Royal Road
Springfield, Virginia 22161
Etats-Unis

ESA/Information Retrieval Service
European Space Agency
10, rue Mario Nikis
75015 Paris
France

The British Library
Document Supply Division
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
Royaume Uni

Les demandes de microfiches ou de photocopies de documents AGARD (y compris les demandes faites auprès du NTIS) doivent comporter la dénomination AGARD, ainsi que le numéro de série de l'AGARD (par exemple AGARD-AG-315). Des informations analogues, telles que le titre et la date de publication sont souhaitables. Veuiller noter qu'il y a lieu de spécifier AGARD-R-nnn et AGARD-AR-nnn lors de la commande de rapports AGARD et des rapports consultatifs AGARD respectivement. Des références bibliographiques complètes ainsi que des résumés des publications AGARD figurent dans les journaux suivants:

Scientifique and Technical Aerospace Reports (STAR)
publié par la NASA Scientific and Technical
Information Division
NASA Headquarters (NTT)
Washington D.C. 20546
Etats-Unis

Government Reports Announcements and Index (GRA&I)
publié par le National Technical Information Service
Springfield
Virginia 22161
Etats-Unis

(accessible également en mode interactif dans la base de données bibliographiques en ligne du NTIS, et sur CD-ROM)

Imprimé par Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ